INSTITUTUL POLITEHNIC IAȘI
Facultatea de Automatică și Calculatoare

LUCRĂRILE
CELUI DE AL III-LEA SIMPOZION

„STRUCTURI, ALGORITMI ȘI ECHIPAMENTE DE CONDUCERE A PROCESELOR INDUSTRIALE

Vol. II

IAȘI
25—26 octombrie 1991

Scanned with OKEN Scanner

LUCRĂRILE

celui de al III-lea Simpozion

„ Structuri, algoritmi și echipamente de conducere a proceselor industriale "

Secția D:

Structuri hardware pentru prelucrarea informațiilor

Secția E:

Structuri software pentru conducere, simulare și proiectare asistată

IAȘI

25 – 26 octombrie 1991

Scanned with OKEN Scanner

Controler de linie telefonica

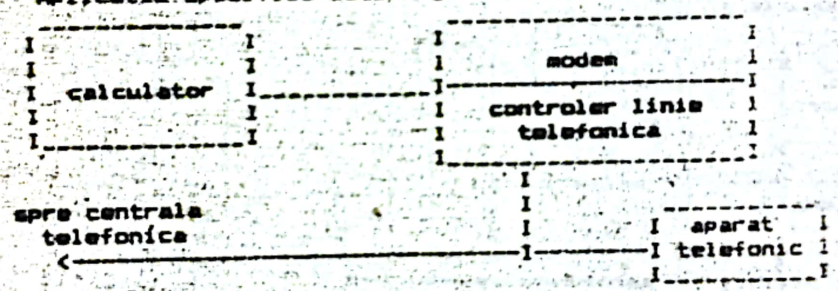
Autori: conf.dr.ing Graur Aurian
as.ing. Potorac Dan-Alin
as.ing. Morosan Tudor

Universitatea "Stefan cel Mare" Suceava

1. Introducere

Conceperea acestui echipament a fost determinata de necesitatea realizarii de transferuri de date intre sistemele de calcul folosind suporturi fizice existente, respectiv liniile telefonice. In prezent legatura intre doi abonati telefonici pentru transmisiuni de date se realizeaza manual sau se folosesc linii telefonice concesionate.

Aplicatia specifica este sugerata in figura de mai jos.



2. Descriere generala

Controlerul de linie telefonica care face obiectul acestei descrieri este destinat stabilirii unei legaturi pentru comunicatiile seriale prin centrala telefonica automata .

Concret , acest echipament are urmatoarele functiuni :

- sesizeaza cererea de stabilire a legaturii .
- avertizeaza , un timp prestabilit , ca urmeaza sa se preia controlul liniei telefonice
- decupleaza un eventual telefon legat la linie
- introduce pe linie o impedanta mare (corespunzatoare starii "telefon inchis"), sau o impedanta mica (corespunzatoare

starii "telefon deschis"
- sesizeaza existenta buclei de curent prin linie (ca informatii asupra integritatii liniei)
- formeaza numarul de telefon al' abonatului care trebuie apelat
- cupleaza modemul la linia telefonica
- detecteaza raspunsul abonatului chemat
- citeste , eventual , starea "detectie purtatoare" de la modem
- detecteaza apelul telefonic .

Modulul functioneaza incorporat oricarui microsistem unde exista posibilitatea accesarii magistralei de date si de adrese , apelarea sa facindu-se ca port , oriunde in spatiul de adresare al microprocesorului . O aplicatie optima consta in asigurarea secventierelor necesare de catre un microprocesor (de exemplu MMN 80 - 'CPU) care contorizeaza intreruperile generate de un circuit de temporizare-numarare (eventual de tipul MMN 80 - CTC) . Pot fi imaginate si alte posibilitati de configurare in cadrul microsistemului .

Numarul ce urmeaza a fi "format" este stabilit prin program.
Resetarea controlerului se face soft sau hard .

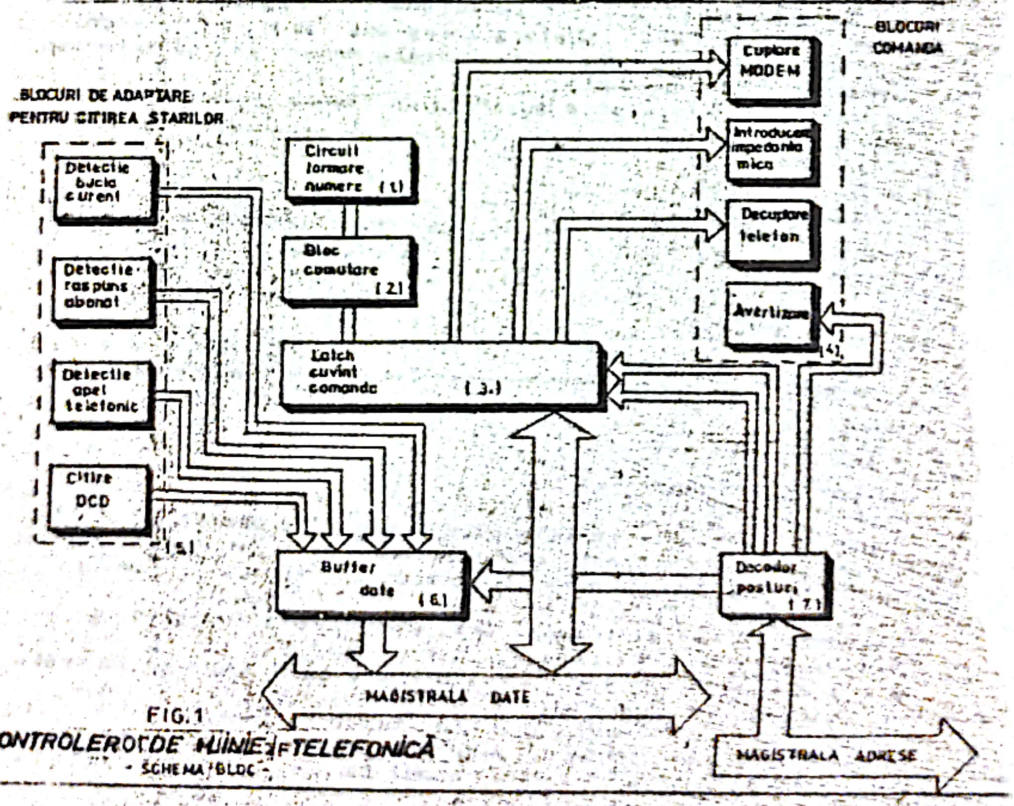


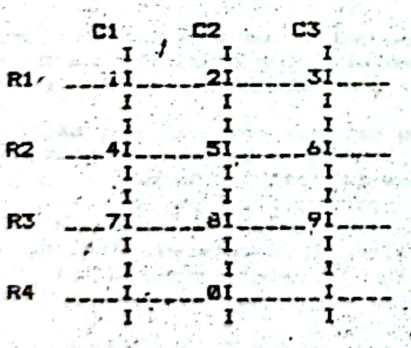
FIG.1
CONTROLEROL DE LINIE TELEFONICĂ
- SCHEMA BLOC -

Principalele blocuri functionale sint prezentate in continuare .

1. Bloc formare numere - este realizat cu circuitul specializat MMN 760 si realizeaza formarea numarului dorit .
2. Bloc comutare - executa inchiderea contactelor necesare pentru formarea unui anumit numar ; este realizat cu multiplexoare analogice integrate .
3. Latch cuvint comanda - retine cuvintul de comanda trimis prin program . Structura acestui cuvint defineste secventa ce urmeaza a fi executata .
4. Blocuri de comanda - realizeaza comanda efectiva a secventei impuse (semnalizare , decuplare telefon , introduce impedanta mica pe linie , cuplare modem) , folosind , cu precadere , releee .
5. Blocuri de adaptare pentru citirea starilor - permit citirea starilor (detectia existentei buclei de curent a centralei , detectie raspuns abonat , detectie apel telefonic etc.)
6. Buffer 3-state - este necesar pentru citirea starilor pe magistrala de date .
7. Decodorul de adrese porturi - decodifica starea magistralei de adrese permitind adresarea porturilor de pe modul .

3. Functionare

Partea cea mai importanta a acestui modul o constituie circuitul specializat MMN 760 (Loop Disconnect Dialler) care asigura intreruperea buclei de curent continuu a centralei de un numar de ori egal cu cifra din numarul abonatului chemat care este "formata" la un moment dat , circuitul asigurind si pauzele corespunzatoare intre cifre . Comanda trimiterii unei anumite cifre se face prin inchiderea contactului corespunzator intr-o matrice de 4 linii si 3 coloane (figura de mai jos).



Tastarea unei anumite cifre se realizeaza fizic prin stabilirea unui contact electric intre linia si coloana corespunzatoare , conform reprezentarii alaturate.

Solutia adoptata in cazul dispozitivului prezentat foloseste doua multiplexoare analogice de tipul " 4 cu 1 " (2 x 1/2 MMC 4051) cu intrarile/iesirile comune legate impreuna . Adresarea multiplexoarelor se face dintr-un latch incarcat de pe magistrala de date a sistemului cu un cuvint de control adecvat . Latch-ul este adresat ca port , la o adresa stabilita prin

Scanned with OKEN Scanner

jumpere, oriunde in spatiul de adresare al porturilor de catre microprocesor. Comenzile necesare sint prescrise prin inscrierea, in mod similar ca mai sus , a unui cuvint de comanda pe 4 biti, activarea oricarui bit determinind lansarea uneia din comenzile asociate , si anume :

1. Decuplarea aparatului telefonic local de la linia telefonica .

2. Conectarea unei impedante reduse (comparabile cu impedanta castii telefonice) pe linie pentru ca centrala sa "sieta" starea "receptor ridicat" necesara pentru trimiterea semnalului de linie libera,(ton) .

3. Cuplarea unui modem la linia telefonica pentru a permite comunicatia seriala de date .

4. Dezinhibarea multiplexoarelor pentru stabilirea contactului electric asociat unei anumite cifre , conform cuvintului de control inscris in latch-ul respectiv .

Activarea si dezactivarea comenzilor , precum si temporizarile necesare (durate de activare , pauze intre comenzi) sint controlate de microprocesor , fara ca aceasta sa insemne ocuparea timpului sau de lucru , el putind sa contorizeze , de exemplu , intreruperile externe generate de un circuit numarator-temporizator.

O facilitate importanta a acestui modul este legata de posibilitatea citirii diferitelor stari :

1. Detectia de linie telefonica activa (detectia semnalului " ton ") da informatii asupra integritatii liniei si permite determinarea momentului cind poate incepe " formarea " numarului .

2. Detectie raspuns abonat chemat (facilitate posibila in cazul centralelor telefonice automate) - sesizeaza momentul cind la abonatul chemat este introdusa o impedanta mica pe linie , adica sesizeaza aparitia starii " receptor ridicat " la abonat .

Cele doua situatii de mai sus sint determinate pe principiul cunoasterii polaritatii curentului in linie prin folosirea a doua optocuploare conectate antiparalel .

3. Detectia apelului telefonic este posibila prin recunoasterea semnalului asociat apelului (trenuri de impulsuri cu frecventa de 25 Hz) si setarea unui bistabil a carui iesire poate fi citita de microprocesor .

4. Detectie purtatoare (sau citirea oricarui alt semnal , optional) permite sesizarea existentei in linie a purtatoarei generata de un modem cuplat la linie la postul chemat .

Citirea acestor stari se face periodic , sau de cite ori este cazul , pe magistrala de date , printr-un buffer 3-state .

Porturile asociate latch-urilor de comanda si control si buffer-ului 3-state de citire a starilor sint adresate printr-un bloc de decodare adrese , realizat cu o cascada de circuite decodificatoare BCD/zecimal .

Exista si o posibilitate de semnalizare optica sau acustica utila pentru cazul avariilor , al imposibilitatii stabilirii legaturii , sau pentru avertizare asupra preluarii controlului liniei telefonice si , implicit , al blocarii unui eventual aparat telefonic .

Dispozitivul poate fi adus oricind in starea initiala prin reset extern sau prin adresare la portul de resetare .

Scanned with OKEN Scanner

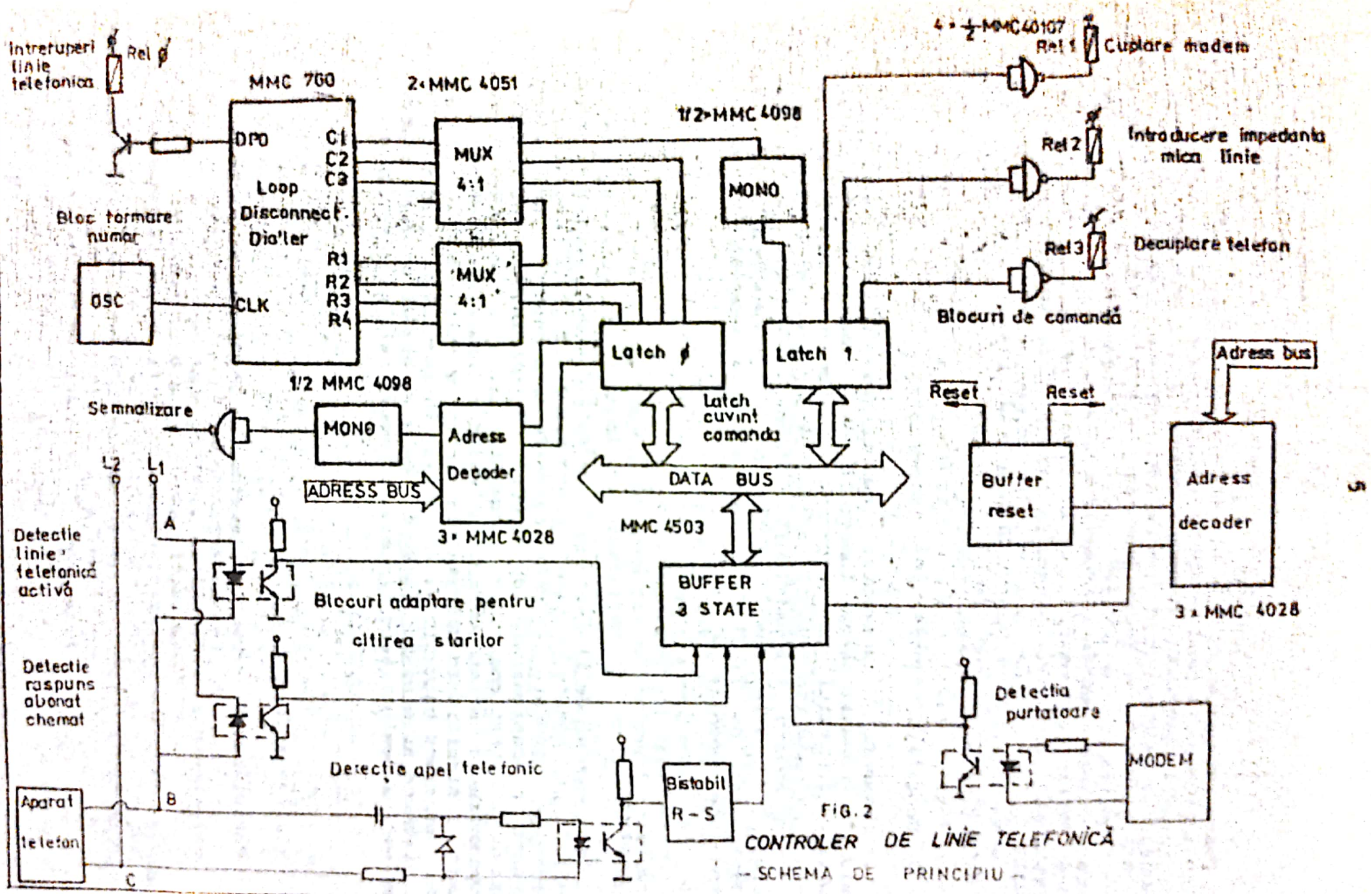


FIG. 2

CONTROLER DE LINIE TELEFONICĂ

– SCHEMA DE PRINCIPIU –

Scanned with OKEN Scanner

Toate intrarile si iesirile modulului sint separate galvanic de linia telefonica prin relee si optocuploare . Pentru realizare s-au folosit circuite integrate MOS (familia MMC 4000) care asigura un consum redus si o mare imunitate la zgomot .

Echipamentul a fost conceput initial ca o extindere a posibilitatilor oferite de microcalculatorul MIND (microcalculator modular cu destinatie speciala si industriale , produs "Microelectronica"), o utilizare eficienta necesitind configuratia minimala alcatuita din modulele MIND UC2 si MIND SIO-CTC . Conceptia modulara si conectica standard permit utilizarea facila in orice microsistem unde exista acces la magistralele de sistem .

4. Concluzii

Echipamentul prezentat realizeaza monitorizarea liniei telefonice, substituind functiile unui aparat telefonic si permitind controlul cu ajutorul calculatorului. Avantajul sau esential consta in faptul ca permite stabilirea de legaturi temporare pentru transmisiuni de date pe linii telefonice comutate (prin centrale telefonice) fara a necesita concesionarea de linii , folosind posibilitatile sistemului national de comunicatie .

Rezumat

Controlerul de linie telefonica este destinat stabilirii de legaturi temporare pentru transmisiuni de date pe linii telefonice comutate (prin centrale telefonice) fara a fi necesara concesionarea de linii , folosind posibilitatile sistemului national de comunicatii. Dispozitivul se caracterizeaza printr-un consum redus , imunitate ridicata la zgomot , fiabilitate sporita (conferite de folosirea circuitelor MOS), separare galvanica a intrarilor si iesirilor , grad inalt de integrare si miniaturizare. El poate fi atasat oricarui sistem unde exista posibilitatea accesarii magistralelor de sistem .

Bibliografie

Postelnicu P. - Linii si sisteme de transmisiuni telefonice
Ed. tehnica, 1984

Holdsworth B. - Microprocessor Engineering
Ed. Butterworths, England, 1987

Rinkenbe Th. - Proiectarea cu circuite MSI si LSI standard
Ed. tehnica, 1988

Ardelean I. - Circuite integrate CMOS, manual de utilizare
s.a. Ed. tehnica, 1986 ,

* - Data Book, "Microelectronica" 1989

Echipament pentru transmisiuni de date

Autori: conf.dr.ing. Graur Adrian
as.ing. Potorac Dan-Alin
as.ing. Morosan Tudor

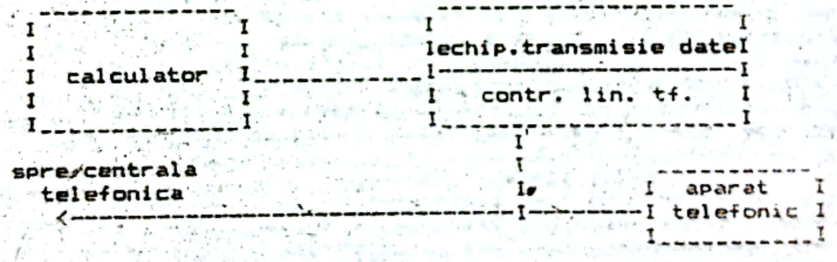
Universitatea "Stefan cel Mare" Suceava

1. Introducere

In proiectarea acestui echipament s-a pornit de la necesitatea practica de a facilita comunicatia intre sisteme de calcul in aplicatii de teleconducere , telegestiune , achizitii de date , comenzi la distanta , transmisiuni teletext , accesare facila a bancilor de date , etc. , folosind posibilitatile oferite de sistemul national de comunicatii la ora actuala . Concret , acest echipament permite transmiterea de date semiduplex (half-duplex) pe linii telefonice comutate sau inchiriate. Stabilirea legaturii intre cei doi abonati telefonici este realizata cu ajutorul unui controler de linie telefonica care nu face obiectul acestei prezentari.

Echipamentul asigura codarea in frecventa a semnalului ce urmeaza a fi transmis , adaptindu-l la banda de frecventa a liniei telefonice. La receptie se realizeaza decodarea FSK corespunzatoare.

O posibilitate de utilizare este cea din figura de mai jos.



2. Descriere generala

Echipamentul prezentat este un modem care realizeaza transmisia seriala , half-duplex , pe linie telefonica, la emisie realizindu-se o codare FSK (modulare), partea de receptie asigurind decodarea corespunzatoare (demodulare) . Modulul poate fi folosit independent sau atasat oricarui microsistem care poseda interfata seriala standard RS 232 , fara a necesita acces

la magistralele de sistem .

Structura de baza contine :

- lantul de emisie
- lantul de receptie
- blocuri auxiliare

Blocurile functionale pot fi urmarite pe schema bloc.

Blocurile functionale

a. Structura lantului de emisie

a.1 Oscilatorul - este o structura clasica de oscilator cu cuart care genereaza tactul necesar obtinerii purtatoarei .

a.2 Divizorul comandat - realizeaza divizarea necesara obtinerii frecventelor asociate semnalelor logice "0" respectiv "1" . Comanda divizorului se face cu semnalul ce urmeaza a fi transmis .

a.3 Numaratorul divizor - genereaza adresele necesare blocului de memorie .

a.4 Memoria ROM - contine , in binar , valoarea esantioanelor pentru generare functiei sinus .

a.5 Convertorul digital-analogic - transforma valoarea binara a esantioanelor din memorie in nivele de tensiune continua , obtinindu-se o forma de unda sinusoidala , cu o rezolutie de 1/256 , numarul de pasi fiind prescris din blocul de numarare-divizare , in functie de frecventa oscilatorului cu cuart si de precizia dorita .

a.6 Filtrul trece-jos - realizeaza "netezirea" semnalului obtinut la iesirea convertorului digital-analogic pentru atacarea liniei cu o forma de unda cit mai apropiata de cea ideala .

a.7 Separatorul de sens - este un bloc analogic care permite separarea sensurilor de emisie respectiv receptie precum si adaptarea de impedanta spre linia de transmisie .

b. Structura lantului de receptie

b.1 Blocul pentru reglarea automata a amplificarii - este o structura analogica care mentine constant nivelul semnalului la iesire compensind atenuarea liniei realizind si-o prima filtrare trece banda .

b.2 Adaptorul - formator - a fost introdus pentru a realiza translarile de nivel necesare precum si transformarea semnalului sinusoidal in semnal dreptunghiular .

b.3 Filrul trece - banda - filtru digital ce rejecteaza semnalele cu frecventa in afara benzii utile .

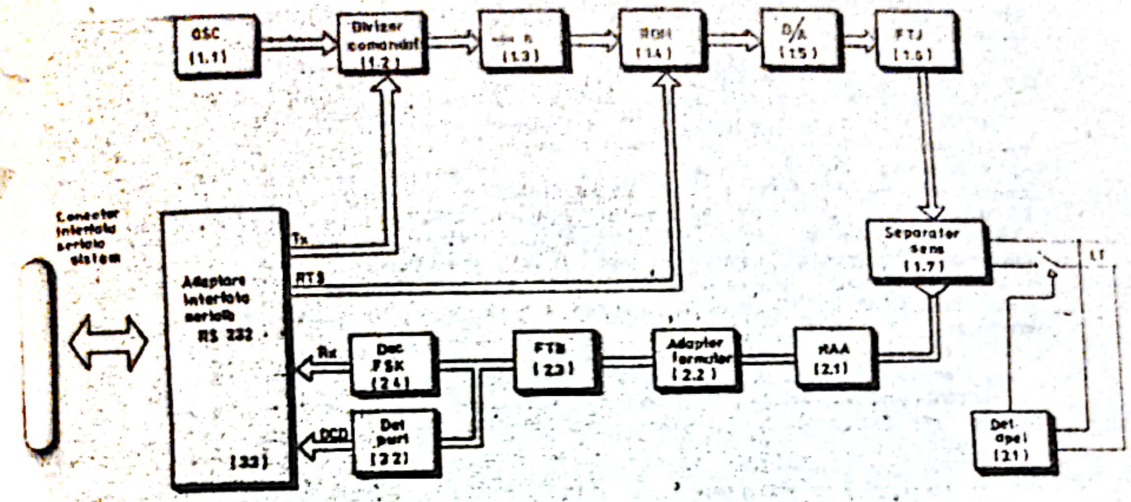
b.4 Decodorul FSK - contine in esenta o bucla PLL cu un indicator de calare , care realizeaza decodarea propriu-zisa .

c. Blocurile auxiliare

c.1 Detectorul de apel telefonic - are o structura care ii permite detectarea apelului telefonic si semnalizarea corespunzatoare prin aprinderea unui LED si , eventual , actionarea unui releu care sa cupleze modemul la linia telefonica.

c.2 Detectorul de purtatoare - sesizeaza existenta purtatoarei si semnalizeaza aceasta prin activarea unui semnal de control si aprinderea unui LED.

Scanned with OKEN Scanner



ECHIPAMENT PENTRU TRANSMISIUNI DE DATE

SCHEMA BLOC

FIG.1

c.3 - Bloc adaptare interfata seriala RS 232 - realizeaza adaptarea conform standardului RS 232 pe partea de comunicatie seriala .

3. Functionare

Pentru transmiterea datelor pe linii de comunicatii este necesara codificarea informatiei , codificare care se face prin modulare in frecventa (FSK - frequency shift keying) . Conform standardelor internationale se asociaza bitului "0" logic frecventa de 2100 Hz iar bitului "1" logic frecventa de 1300 Hz .

Obtinerea acestor frecvente se face prin divizari succesive, pornind de la frecventa oscilatorului cu cristal de cuart .

Notam cu :

- f_0 - frecventa oscilatorului cu cristal de cuart
- n_0 - factorul de divizare al divizorului comandat , asociat bitului "0"
- n_1 - factorul de divizare al divizorului comandat , asociat bitului "1"
- N - factor de divizare al divizorului programabil
- f_0 - frecventa la iesire (emisie) asociata bitului "0"
- f_1 - frecventa la iesire (emisie) asociata bitului "1"

Urmarind structura lantului de emisie si avind in vedere cascadarea circuitelor de divizare , frecventele de iesire au valorile :

$$f0 = F0 / (n0 \times N)$$
$$f1 = F0 / (n1 \times N)$$

Valorile f0 si f1 sint standardizate . Factorul de divizare al divizorului progamabil , N , trebuie sa fie egal cu numarul de esantioane inscris in memoria ROM . Divizorul programabil furnizeaza, pe maximum 11 biti de adresa (disponibili la o memorie 2716 de 2 ko) , in binar, valori numerice crescatoare de la 0 la N . La adresele respective (succesive) in memorie sint inscrise , in binar , pe 8 biti , valorile esantioanelor pentru generarea functiei sinus . Aceste secvente binare sint convertite intr-o tensiune analogica de un convertor digital-analogic de 8 biti realizat cu circuitul integrat DAC 08 . Valoarea unui esantion este :

$$e(p) = k \times (2exp8)/2 \times [1 + \sin(2n/N) \times p] ,$$

unde $0 < p < N$, n = nr. natural

k = constanta de proportionalitate dependenta de domeniul maxim de variatie a tensiunii la iesirea convertorului digital-analogic

S-a optat pentru urmatoarele valori :

F0 = 7,2072 MHz n0 = 13
N = 264 n1 = 21

Cu aceste valori , conform relatiilor anterioare , se obtin frecventele dorite f0 = 2100 Hz , respectiv f1 = 1300 Hz .

Separarea sensurilor de emisie si receptie se face intr-un bloc analogic realizat cu amplificatoare operationale . Separarea galvanica si adaptarea de impedanta spre linia de transmisie este asigurata de un transformator .

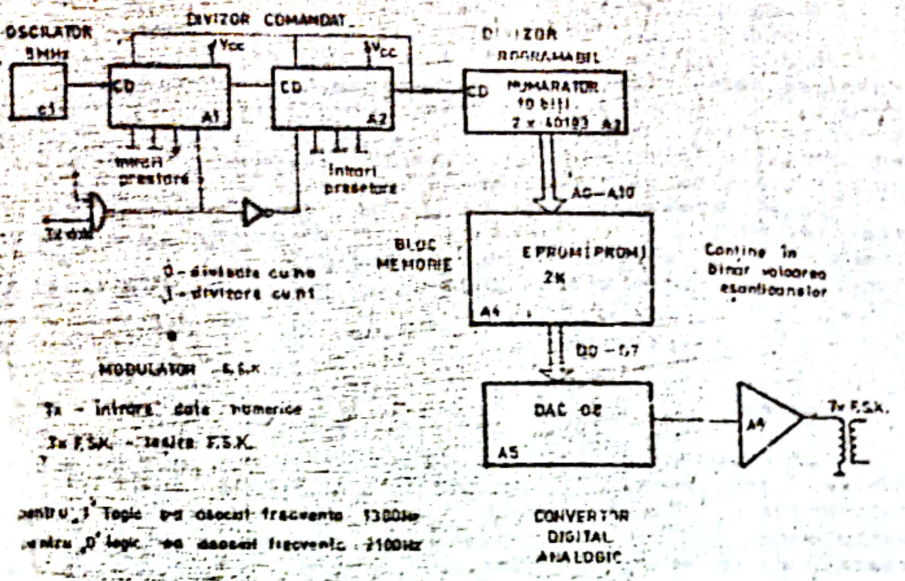
La receptie trebuie sa se discearna intre cele doua frecvente f0 si f1 . Pentru aceasta semnalul sosit pe calea de comunicatie este trecut printr-un bloc de filtrare compus din doua filtre trece-banda legate in paralel , avind frecventele centrale f0 , respectiv f1 si o latime de banda suficient de mica pentru a minimiza influenta semnalelor parazite . Semnalul astfel prelucrat este aplicat la intrarea unui circuit de tip PLL (phase locked loop - bucla cu [illegible]are pe faza) avind banda de calare centrata pe frecventa f0 = 2100 Hz. Cu un circuit indicator de calare se poate sesiza existenta in linie a frecventei f0. Lipsa acestui semnal este interpretata prin inexistenta in linie a unui semnal util , sau prin existenta in linie a frecventei f1 daca a fost activat si semnalul DCD (Data Carrier Detect - detectie de purtatoare) . Semnalul DCD este generat de un monostabil retrigerat continuu de unul din semnalele f0 sau f1 .

Pentru asigurarea unui nivel constant la intrarea in circuitul PLL a fost prevazuta o reactia negativa pe amplificatorul de intrare , de tip RAA (reglaj automat al amplificarii) .

Echipamentul prezentat este realizat cu componente indigene. S-au folosit circuite integrate MOS din familia MMC 4000 (numaratoare presetabile 40193, circuit PLL 4046, etc.), memorie ROM tip MMN 2716 de 2 ko, convertor digital-analogic DAC 08, partea analogica fiind realizata cu AO tip BM 324.

Scanned with OKEN Scanner

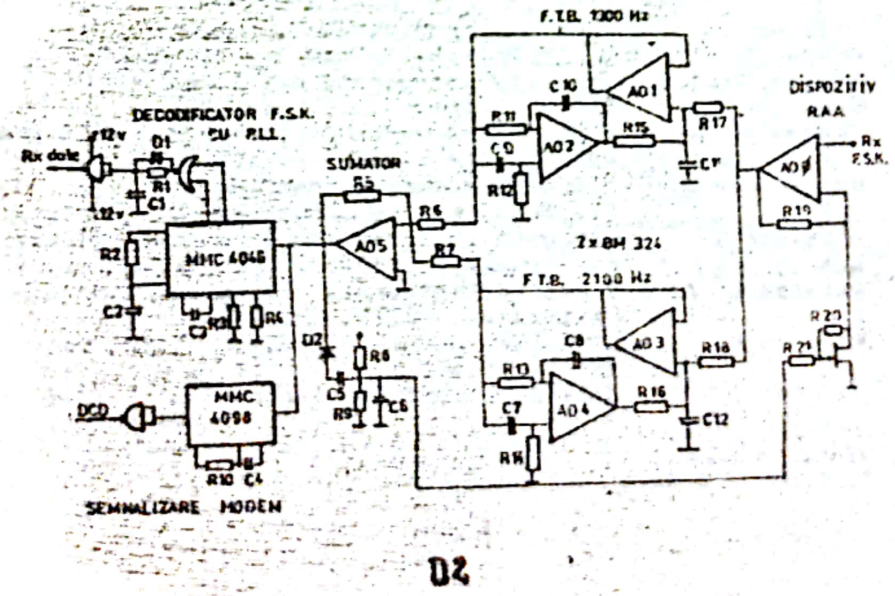
FIG. 2



PARTEA DE EMISIE

PARTEA DE RECEPTIE

FIG. 3



Scanned with OKEN Scanner

4. Concluzii

Gradul ridicat de integrare, minimizarea partii analogice in favoarea celei numerice , utilizarea circuitelor MOS, precum si solutiile tehnice adoptate atribuie acestui echipament caracteristici specifice : imunitate ridicata la zgomot si perturbatii, consum redus, fiabilitate sporita, depanare usoara cu posibilitate de auto test, pret scazut.

Acest modem a fost conceput initial ca o extindere a familiei MIND (microcalculator modular cu destinatie speciala si industriala ,' produs "Microelectronica"). Structura modulara si conectica standard permit insa conectarea facila la orice sistem.

Rezumat

Echipamentul pentru transmisiuni de date este un modem half-duplex care permite transmisia seriala a datelor pe linii telefonice. Codarea FSK se realizeaza prin generarea purtatoarei cu un secventiator cu memorie PROM, secventiator atacat cu un semnal de ceas cu frecventa dependenta de valoarea logica a bitului ce se transmite la un moment dat. La receptie, decodarea se face printr-un indicator de calare atasat unui circuit PLL. Solutia tehnica folosita a permis minimizarea 4volumului de componente, corelat cu un consum scazut si o mare imunitate la zgomot.

Bibliografie

Millman I. - Microelectronics
Grabel A. Ed. McGrow Hill, S.U.A. , 1987
Stefan Bh.M. - Circuite integrate digitale
s.a. E.d.p. , 1983
Blakeslee Th. - Proiectarea cu circuite MSI si LSI standard
Ed. tehnica , 1988
Holdsworth B. - Microprocessor Engineering
Ed. Butterworths , England , 1987
Ardelean I. - Circuite integrate CMOS, manual de utilizare
s.a. Ed. tehnica , 1986
Vatasescu A. - Circuite integrate liniare, manual de
s.a. utilizare, vol.1 , Ed. tehnica , 1979
Rapeanu R. - Circuite integrate analogice, catalog
Ed. tehnica , 1983
- Data Book: , "Microelectronica" , 1989

Scanned with OKEN Scanner

Interfata specializata pentru achizitii de marimi analogice

Autorii conf.dr.ing. Graur Adrian
as.ing. Potorac Dan-Alin
as.ing. Morosan Tudor

Universitatea "Stefan cel Mare" Suceava

1. Introducere

Interfata specializata pentru achizitii de marimi analogice este un echipament care realizeaza selectarea sursei de semnal si conversia digitala a marimii respective, fiind controlata de un microprocesor de 8 biti. Ea poate fi atasat oricarui microsistem care are magistralele accesibile. Aplicatia fundamentala pentru care a fost conceput acest echipament se incadreaza in categoria aparaturii medicale si este un aparat pentru termografie de contact.

Aparatul pentru termografie de contact pune la dispozitie unui sistem de calcul cu posibilitati grafice color (din familia PC) datele necesare obtinerii unei harti termice a organismului uman , harta ce permite depistarea timpurie a diferitelor afectiuni .

In cazul utilizarii unui sistem MIND (microcalculator modular cu destinatie speciala si industriala produs de "Microelectronica" Bucuresti) structura minimala cuprinde modulele: UC2 (unitate centrala) , SIO-CTC (transmisie seriala) si cele doua module specializate care fac obiectul acestei prezentari , AD12 si MX .

Modulul AD12 - convertor analogic-digital pe 12 biti . Realizeaza conversia propriu-zisa sub controlul unui soft adecvat .

Modulul MX - multiplexor analogic pentru 64 (32) canale de intrare . Permite selectarea canalului ce urmeaza a fi "citit" .

Scanned with OKEN Scanner

2. Modulul de conversie analog-digitala pe 12 biti

2.1. Descriere generala

Modulul AD12 este un convertor analogic-digital pe 12 biti cu reactie functionind pe principiul compararii tensiunii de convertit cu o tensiune generata de un convertor D/A de 12 biti comandat cu un latch prescalat prin program . In functie de soft-ul asociat conversia se poate face dupa metoda "cu incrementare (decrementare)" sau "cu aproximatii succesive". Structura hard a acestui modul este erganizata in jurul unui circuit PIO care permite controlul resurselor hardware de catre microprocesor .

Blocurile functionale si functiunile lor sint prezentate in continuare .

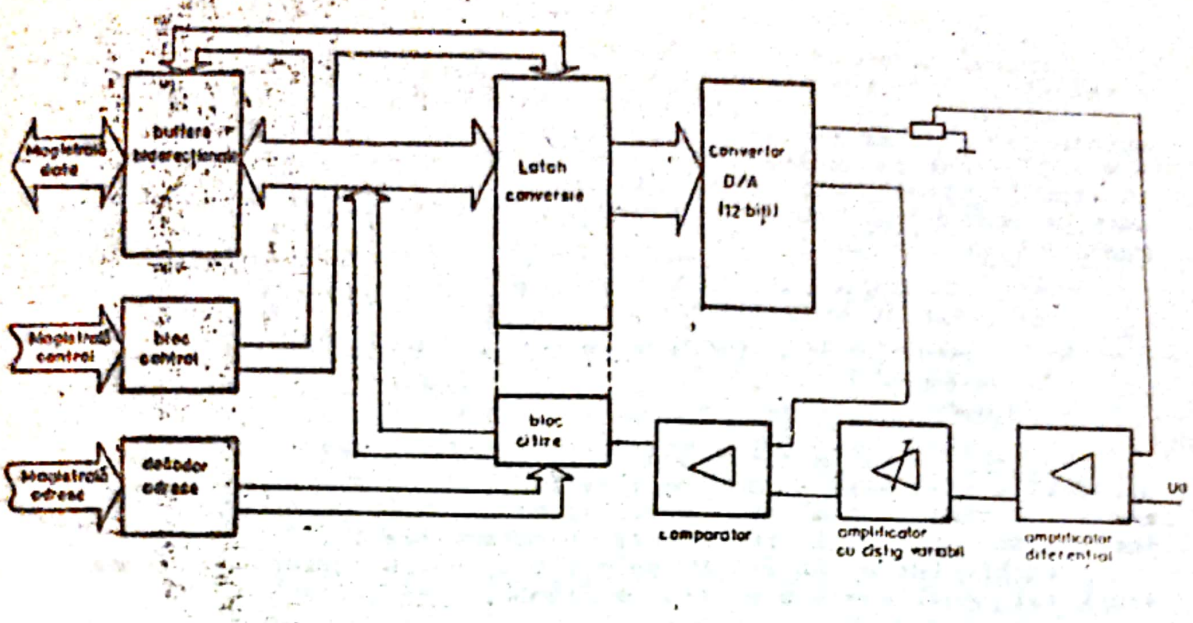
* latch conversie - retine 12 biti trimisi prin program , necesari blocului de conversie D/A
* bloc conversie D/A - este realizat cu un convertor specializat pe 12 biti care genereaza tensiunea analogica corespunzatoare , tensiune aplicata blocului de comparare .
* amplificator diferential de intrare - amplifica x 1 diferenta dintre tensiunea de referinta si tensiunea de convertit .
* amplificator cu cistig variabil - amplifica x 16 tensiunea diferenta generata de blocul anterior .
* comparator - sesizeaza printr-un nivel logic corespunzator relatia de ordine dintre valoarea corespunzatoare tensiunii de intrare si tensiunea prescrisa prin program .
* blocul de citire - permite citirea starii comparatorului (3 - state) .
* buffer-e bidirectionale - asigura bidirectionalitatea si translarile de nivel necesare pentru accesul pe magistrala de date .
* bloc control - realizeaza adaptarea de nivel pentru semnalele de control .
* decodor adrese - decodifica starea magistralei de adrese permitind adresarea in spatiul porturilor pentru cicuitele tip latch si blocul de citire (PIO) .

2.2. Functionarea convertorului

Structura hard a modulului de conversie are la baza un circuit MMN 80 - PIO care contine doua porturi de 8 biti . Portul A si 1/2 din portul B asigura 12 biti la intrarea circuitului specializat de conversie digital - analogica pe 12 biti (AD DAC - 80) . Ceilalti 4 biti sint folositi conform schemei de mai jos :

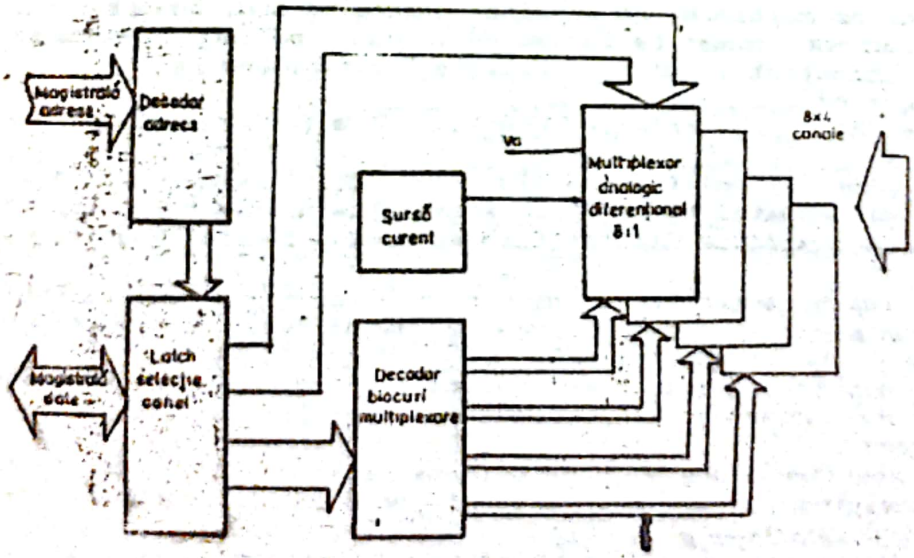
D7 A - D0 A (PIO) = D11 - D4 (AD DAC - 80)
D7 B - D4 B (PIO) = D3 - D0 (AD DAC - 80)
D3 B (PIO) = neutilizat
D2 B (PIO) = neutilizat
D1 B (PIO) = neutilizat
D0 B (PIO) = intrare pentru citirea starii comparatorului

Scanned with OKEN Scanner



MODUL DE CONVERSIE ANALOG- DIGITALA
PE 12 BITI
SCHEMA BLOC

MODUL MULTIPLEXARE
Schema bloc



Scanned with OKEN Scanner

Pentru conversiile relativ rapide este indicat a se utiliza principiul aproximatiilor succesive .

Pentru conversia in zecimal a rezultatului hexazecimal obtinut se va avea in vedere capatul de scala pentru AD DAC 80 (5 V) , faptul ca iesirea convertorului este invers proportionala cu codul hexazecimal aplicat la intrare si numarul de biti pe care se obtine rezultatul (eventual in urma unor calcule) . Se poate folosi formula :

$$Nzec = Nhex \times c / (2^{b} - 1)$$

b = numar de biti pe care se obtine rezultatul (12 , de exemplu).
c = capat de scala (5 V)

Factorul de amplificare al amplificatorului cu castig variabil a fost ales 16 pentru a facilita eventualele calcule in binar (deplasarile stinga , dreapta sint echivalente cu inmultirea , respectiv impartirea cu puteri ale lui 2) .

Posibilitatea AD DAC 80 de a lucra in cod zecimal nu ofera facilitati suplimentare si reduce rezolutia de conversie .

Este posibila utilizarea lucrului in intreruperi cu PIO pentru sesizarea schimbarii starii comparatorului , de exemplu .

Reglajele necesare sint: offset convertor D/A, liniaritate convertor D/A, amplificare semnal analogic si tensiune referinta.

Pasul conversiei este de 5V / 2exp12 = 1,22 mV

Pentru o amplificare egala cu 16 modulul asigra o rezoltie in tensine de 76,2 uV si o rezolutie in temperatura de 7.62 miligrade Celsius.

3. Modulul multiplexor

3.1. Descriere generala

Modulul de multiplexare permite selectarea unui anumit canal analogic dintr-un numar de 32 sau 64 . canal pe care se trimite un curent constant , si se " citeste " o tensiune pe o linie analogica comuna .

Schema bloc , relativ simpla , este prezentata in continuare :

* blocul multiplexoarelor - constituie elementul principal al acestui modul si este alcatuit din 4 sau 8 multiplexoare analogice diferentiale cu cite 8 canale , de tipul MMC 4097 .
* latch selectie canal - retine cuvintul de comanda trimis prin program , cuvint care , decodificat , determina selectia unui anumit canal . Este realizat cu MMC 4508 .
* decodor blocuri multiplexare - decodifica partial cuvintul de comanda si dezinhiba chip-ul multiplexorului corespunzator .
* decodor adrese - decodifica starea magistralei de adrese asigurind adresarea in spatiul porturilor . Decodoarele folosite sint MMC 4028 .
* sursa curent - asigura curentul constant necesar a fi

Scanned with OKEN Scanner

trimis canalelor selectate .

3.2. Functionarea multiplexorului

Cuvintul de comanda trimis prin program la adresa care determina selectarea si inscrierea latch-ului (adresa stabilita prin jumperi) are structura de mai jos :

D7 = activ sus , determina comutarea sursei de curent
D6 = activ sus , valideaza decodificatorul blocurilor de multiplexare
D5 - D3 = dezinhiba chip - ul multiplexorului ce urmeaza a fi selectat
D2 - D0 = selecteaza canalul de pe fiecare multiplexor

Selectarea latch - ului se face ca port , clasic .

Pentru selectarea canalului 0 , de exemplu , structura cuvintului de comanda trimis este :

D7	D6	D5	D4	D3	D2	D1	D0	
1	1	0	0	0	0	0	0	0c0h

Sursa de curent este realizata cu doua tranzistoare in montaj " oglinda de curent ". In timpul cit nu este folosita sursa de curent este comutata pe o rezistenta care simuleaza impedanta sarcinii pentru a reduce oscilatiile de comutare .

Traductoarele folosite in aplicatia "aparat pentru termografie de contact" sint BM 135 (LM 135)(senzori de precizie pentru determinarea temperaturii in gama -55 , +150 grade Celsius , cu o sensibilitate de 10 mV/grad , liniare in toata gama). Polarizarea traductoarelor se face la curent constant (intre 400 uA si 5 mA) iar caderea de tensiune pe ele ofera informatii despre marimea temperaturii .

Impedanta dinamica scazuta (1 ohm) determina o mare imunitate la zgomot chiar in conditiile conectarii traductoarelor cu conductoare relativ lungi (1 - 3 m) .

*

Soft-ul adoptat asigura functiile de monitor , selecteaza periodic traductorul ce trebuie citit , face calibrarea traductoarelor , realizeaza conversia si executa unele calcule (medieri , transformarea unitatilor de masura , etc.) . Datele obtinute sint preluate de un calculator din familia PC-urilor care face prelucrarile ulterioare dorite (deseneaza "harta termica" propriu-zisa , de exemplu).

Metoda de diagnosticare care sta la baza aparatului pentru termografie de contact constituie obiectul preocuparii mai multor colective medicale din tara noastra .

4. Concluzii

Modulele care alcatuiesc interfata specializata pentru achizitii de marimi analogice, atasate unui microsistem pe 8 biti, permit obtinerea unei structuri capabile sa "citeasca"

Scanned with OKEN Scanner

temperatura cu o precizie de 0.0076 grad Celsius de la un numar de 64 (32) de traductoare (expandabil de un numar de ori egal cu numarul porturilor disponibile in microsistem) . Aceste facilitati fac posibila obtinerea unei harti termice a organismului uman din analiza careia se pot depista afectiunile maligne.

Flexibilitatea structurii prezentate face posibila implementarea in diverse alte aplicatii unde este necesara "citirea" unor marimi analogice relativ lent variabile in timp (temperaturi, presiuni, etc.) cu o precizie foarte ridicata.

Rezumat

Interfata specializata pentru achizitii de marimi analogice cuprinde un modul de conversie analogic-digitala pe 12 biti si unul sau mai multe module multiplexoare pentru selectarea canalului analogic. Conversia este controlata soft si se face prin comparatia continua a tensiunii de convertit cu tensiunea generata de un convertor D/A pe 12 biti. Cu ajutorul unui latch si a logicii de decodare asociate se selecteaza, prin intermediul unui multiplexor analogic, o anumita sursa de semnal. Aplicatia de baza este termografia umana de contact, dar interfata poate fi utilizata pentru orice achizitii de semnale analogice, cu adaptari minime.

Bibliografie

Holdsworth B. - Microprocessor Engineering
Ed. Butterworths , England , 1987
- National Semiconductor
catalog, 1983
- Institute of Microelectronics, catalog
Bulgaria, 1988
Rapeanu R. - Circuite integrate analogice, catalog
Ed. tehnica , 1983
Ardelean I. - Circuite integrate CMOS, manual de
utilizare, Ed.tehnica, 1986
- Data Book, "Microelectronica" , 1989

Scanned with OKEN Scanner

ALGORITM ŞI CIRCUIT DEDICAT DE ÎMBUNĂTĂŢIRE A EFICIENŢEI PROCESULUI DE CORECŢIE ÎN TRANSMISIILE SINCRONE CU PONDERE CONSTANTĂ

Autor: As.ing. Rezeanu Ştefan-Cristian, I.P.Iaşi

1. Generalităţi

În cadrul Sistemului mondial de pericol şi siguranţă pe mare - GMDSS (Global Maritime Distress and Safety System), o importantă fracţiune din volumul legăturilor radio este alocată telegrafiei de bandă îngustă - NBDPT (Narrow-Band Direct-Printing Telegraphy), care utilizează moduri de lucru cu detecţia şi corecţia erorilor ([3]). Pentru exemplificări ne vom fixa asupra comunicaţiilor radiotelex, care utilizează transmisii sincrone în cod cu 7 elemente de semnal cu pondere constantă - în combinaţia de semnale unitare raportul dintre cele două stări semnificative fiind întotdeauna 4/3. Tipul de modulaţie utilizat este FSK, notaţiile CCIR fiind:

- pentru polaritatea START: A în cod CCITT nr. 2 (Baudot) şi B (corespunzător frecvenţei superioare) în codul cu 7 elemente de semnal cu pondere constantă;
- pentru polaritatea STOP: Z, respectiv Y (asociat frecvenţei inferioare).

Astfel, combinaţia care reprezintă 0 (în mod literă) sau 9 (în mod cifră) va apare AAAZZ în cod Baudot şi BYYYBBB în cod cu 7 elemente de semnal cu pondere constantă (4B/3Y).

Detecţia erorilor presupune verificarea ponderii cuvintelor de cod recepţionate, în timp ce corecţia se efectuează în mod diferit în funcţie de modul de operare:

- modul semiduplex ARQ (Automatic Retransmission reQuest) utilizează în acest scop retransmisia caracterelor la cererea corespondentului, pînă la recepționarea lor cu pondere corectă (sau întreruperea legăturii datorită condițiilor improprii);
- modul unidirecțional FEC (Forward Error Correction) recurge la emisia cu diversitate în timp, fiecare caracter fiind transmis de două ori (DX - transmisie directa, RX - retransmisie), existînd subclasa FEC-selectiv în care polaritatea semnalului de date este inversată (ponderea devenind 3B/4Y); corecția se efectuează comparînd caracterele RX și DX; dacă unul dintre ele are pondere incorectă, celălalt este imprimat; dacă ambele au pondere incorectă, sau corectă dar caracterele sînt diferite, se imprimă "spatiu".

2. Algoritmul propus

Metodele de corecție expuse prezintă dezavantajul că pot mări uneori nejustificat durata legăturii (ARQ) - putîndu-se chiar lua decizia întreruperii acesteia - sau sînt tipărite mesaje cu multe spații (FEC) în condiții de recepție zgomotoasă.

Algoritmul suplimentar de corecție a erorilor de demodulare primară datorate unei recepții zgomotoase, propus în continuare, necesită sesizarea tranzițiilor nedorite (parazite) ce apar în fluxul de date într-o fereastră de timp centrată pe momentul preluării datei; coroborînd aceste informații (în modul de operare FEC) cu datele de pe pozițiile RX si DX eșantionate de tactul refăcut, se poate obține îmbunătățirea eficienței procesului de corecție.

În diagrama de impulsuri ilustrativă din fig. 1 (unde notațiile corespund schemei bloc din fig. 2), s-a presupus că este folosit codul cu 7 elemente de semnal cu pondere constantă ([3]), emițîndu-se caracterul "0" (sau "9") în mod de lucru FEC-selectiv. Vom presupune, de asemenea, că sincronizatorul de bit a depășit faza de achiziție, tactul refăcut (4 C) fiind corect fazat cu frontul negativ în imediata vecinătate a momentelor în care pot apare tranziții în fluxul de date primar demodulate.

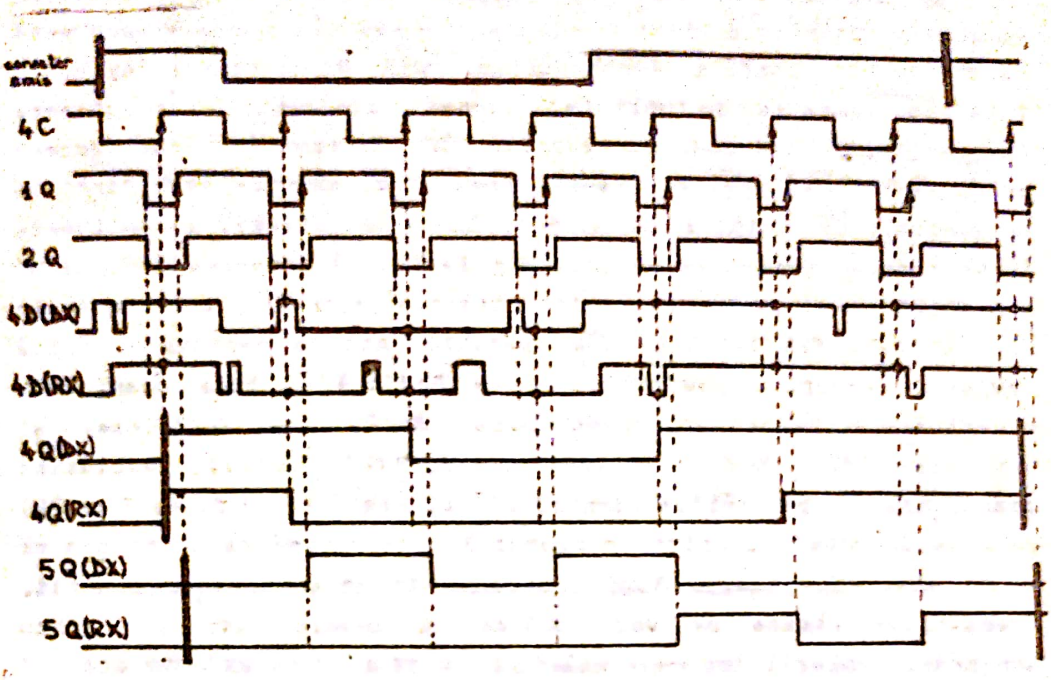


Fig. 1: Forme de undă ilustrative pentru algoritmul propus.

Caracterul este recepționat o dată în transmisie directă (4D-DX) si a doua oară in retransmisie (4D-RX), dar afectat de zgomot, prin scurte impulsuri parazite. Suplimentar fereastrei de timp (1 Q) centrate pe momentul preluării datei (moment marcat de frontul pozitiv al tactului refăcut, unde este maximă probabilitatea unei interferențe intersimbol minime), circuitul dedicat - descris la paragraful 3 - necesită și o fereastră de timp asimetrică, cu frontul crescător întîrziat (2 Q). Frontul crescător al ferestrei centrate (1 Q) va marca momentul în care se decide dacă pe durata ei au existat sau nu tranziții în fluxul de date primar demodulate, în timp ce frontul crescător al tactului refăcut decide asupra valorii logice a biților succesivi emiși (starea în care găsește acest front datele 4D-DX si 4D-RX este marcată cu punct în fig. 1).

Am presupus că a fost emis caracterul avînd, în ordine, stările semnificative 1000111 (YBBBYYY, deci cu pondere 3B/4Y, corectă în situația lucrului în mod FEC-selectiv).

Scanned with OKEN Scanner

Se observă că în condiții de zgomot se recepționează 1100111, cu pondere incorectă 2B/5Y (4Q-DX) si 1000011, cu pondere incorectă 4B/3Y (4Q-RX). Conform recomandărilor CCIR, se va tipări "spațiu", luîndu-se decizia de caracter eronat recepționat. În absența informațiilor privind tranzițiile în fereastră (date notate prescurtat DTF), deși se observă imediat că stările semnificative cu numerele de ordine 1, 3, 4, 6 si 7 au fost identic recepționate în cele două cuvinte (1 00 11), din faptul că ponderea trebuie să fie 3B/4Y, nu se poate concluziona decît că s-ar fi putut transmite fie 1100011, fie 1000111, fără posibilitate de decizie imediată (evident, exceptăm posibilitatea de decizie pe baza examinării ulterioare a cuvintelor recepționate). Dacă, însă, se folosesc și cuvintele DTF (5Q-DX: 0101000 si 5Q-RX: 0000101), excluzînd tranzițiile de pe stările identic recepționate (4 pe DX si 7 pe RX) pe considerentul că există o probabilitate maximă ca tranziția să nu fi afectat valoarea logică corectă în momentul eșantionării, tranzițiile rămase ne vor conduce la ambele cuvinte - prin schimbarea valorii logice a stării 2 pe DX si 5 pe RX - la același caracter, cu pondere corectă 3B/4Y: chiar caracterul emis 1000111.

3. Circuitul utilizat

Circuitul prezentat la nivel de schemă bloc în fig. 2, permite utilizarea metodei suplimentare de corecție descrise în paragraful anterior prin memorarea existenței unor tranziții parazite a datelor în momente apropiate celui în care acestea se preiau. Suplimentar, prin eventuala contorizare a stărilor semnificative pe durata cărora au apărut tranziții în fereastra generată, circuitul propus poate oferi și informații asupra stadiului procesului de calare a sincronizatorului de bit, permițînd controlul vitezei de calare a acestuia (uzual în sensul scăderii ei după efectuarea rapidă a achiziției).

Circuitul folosește ca intrare datele primar demodulate, care atacă atît blocul de memorare a datei 4 cît și detectorul de tranziții al sincronizatorului de bit, presupus a furniza scurte impulsuri pozitive la fiecare tranziție din fluxul de date.

Scanned with OKEN Scanner

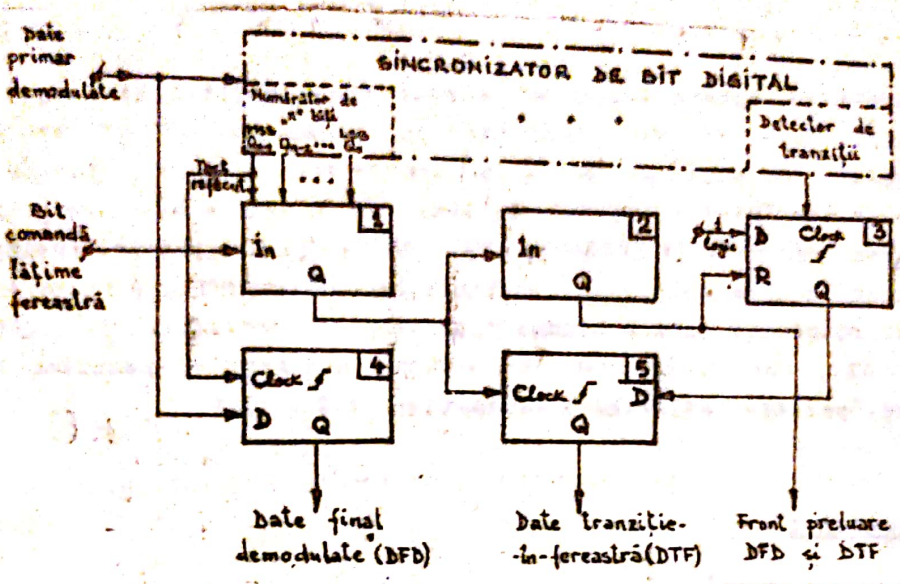


Fig. 2: Schema bloc a circuitului propus.

Numărătorul divizor de frecvență prin "2^n", furnizează tactul refăcut (4 C), al cărui front pozitiv eșantionează datele de la intrarea blocului 1. Numărătorul pe "n" biți și detectorul de tranziții sînt uzuale în sincronizatoarele de bit digitale ([1], [2], [4]), numărătorul fiind obligatoriu; dacă topologia de sincronizator de bit nu conține blocul detector de tranziții, acesta va fi realizat separat, fiind obligatoriu în cazul soluției de circuit propuse.

Ieșirile numărătorului atacă generatorul de fereastră de lățime variabilă 1, lățimea ferestrei fiind presupusă cu două valori prestabilite, prescrise prin intermediul bitului de comandă a lățimii ferestrei. În fereastra generată (pe durata stării 0 logic - 1 Q) se va analiza prezența unor tranziții nedorite în fluxul de date primar demodulate.

Blocul de întîrziere al frontului pozitiv al ferestrei 2, furnizeaza blocului detector de tranziții în fereastră 3, semnalul de reset (2 Q), permițînd astfel blocului de "memorare tranziție în fereastră" 5, memorarea - în momentul frontului pozitiv al ferestrei (1 Q) - existenței a cel puțin unei tranziții pe durata fereastrei.

Existența pe durata ferestrei a unui front pozitiv la intrarea blocului detector de tranziții în fereastră (3C-DX pentru transmisia directă și 3C-RX pentru retransmisie) , provoacă trecerea nivelului 1 logic de la intrarea de date a blocului 3, la intrarea de date a blocului 5 (5D-DX și respectiv 5D-RX). Caracterele formate din datele final demodulate DFD (4Q-DX și 4Q-RX) și respectiv datele asupra tranziției în fereastra DTF (5Q-DX și 5Q-RX), vor fi preluate în vederea analizei si deciziei la frontul pozitiv întîrziat al ferestrei (2 Q).

4. Concluzii

Circuitul propus oferă informații rapide asupra stadiului procesului de calare, permițînd, prin generarea ferestrei variabile de analiză a prezenței tranzițiilor și utilizarea algoritmului de corecție descris, demodularea cu randament îmbunătățit a datelor recepționate; se scurtează durata legăturii prin micșorarea numărului cererilor de repetiție (ARQ), respectiv a numărului de spații tipărite în urma deciziei de "caracter eronat recepționat" (FEC).

BIBLIOGRAFIE:

1. BRANA CODRUȚ, BRANA VALERIA: "Transmiterea informației numerice - vol. II", Editura Tehnică București 1970, Cap. 9 - "Sincronizarea de bit (sinfazarea)", pag. 79-83;
2. GREEN W. T. Jr, KEAN BILL: "Digital Phase-Locked Loops move into analog territory" - Electronic Design, March 31, 1982, pag. 95-100;
3. "Recommandations et rapports du CCIR, 1986", volume VIII-2, Geneve, 1986, ISBN 92-61-02782-2 - Recommandation 490, pag. 76-77 si Recommandation 625, pag. 5-59;
4. SMITH [illegible]: "Modern Communication Circuits" , Mc Graw-Hill 1986, parag. 8.7 - "Digital Phase-Locked Loops", pag. 324-326.

Scanned with OKEN Scanner

CIRCUIT DE INTRODUCERE OPTIMĂ A CORECȚIEI DE FAZĂ, AFERENT UNUI SINCRONIZATOR DE BIT DIGITAL CU URMĂRIREA TRANZIȚIILOR

Autor: As.ing. Rezeanu Ștefan-Cristian, I.P.Iași

1. Generalități

Circuitele sincronizatoare de bit pentru comunicații sincrone permit obținerea în receptor a unui semnal de referință în concordanță de fază cu tactul folosit la emisia datelor (valoarea "f_c" a frecvenței tactului fiind cunoscută). Refacerea cu precizie a fazei tactului de emisie, permite eșantionarea fluxului de date primar demodulate la momente de timp care să minimizeze interferența intersimboluri (conform unuia dintre criteriile lui Nyquist), maximizînd astfel probabilitatea de decizie corectă asupra valorii logice a bitului emis.

Sincronizatoarele de bit cu urmărirea tranzițiilor extrag informația de sincronizare direct din semnalul de date, acesta din urmă trebuind, deci, să prezinte un număr suficient de tranziții. Sincronizatoarele de bit digitale fac aproape în exclusivitate apel la așa-numita metodă de reglaj, care constă în corecția periodică a fazei tactului generat local, prin introducerea unui salt de fază fix la fiecare tranziție a semnalului de date recepționat.

Necesitînd rezolvarea problemei de ajustare a fazei unui semnal generat local, configurația care s-a impus în sincronizarea de bit digitală este cea de DPLL - buclă digitală cu calare de fază ([4]), unul dintre cele mai utilizate circuite integrate de acest tip fiind 54 LS 297 (descris pe larg în [3]).

Din punct de vedere principial, corecția - prin salt - a fazei tactului generat local de un numărător (care înlocuiește în buclele PLL digitale oscilatorul controlat în tensiune), este realizată prin comanda de scădere sau introducere de impulsuri la intrarea de clock a acestuia, de la un oscilator cu ieșiri în opoziție de fază. Implementarea la nivel de schemă bloc, în referința bibliografică [2], a acestui principiu (preluat din [1]), este ilustrată în fig. 1; apare evident dezavantajul unei complexități structurale deosebite.

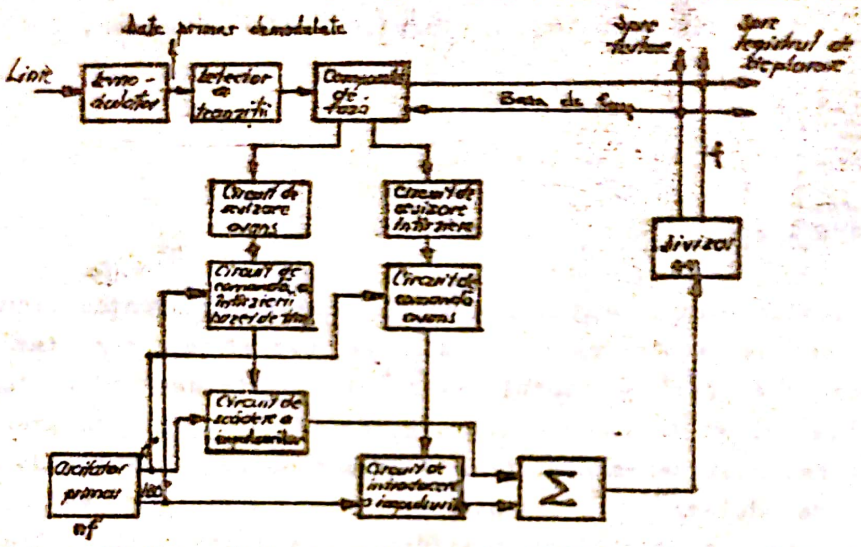


Fig. 1: Schemă bloc de sincronizator cu corecție numerică ([2])

2. Principiul propus

Lucrarea propune o soluție de circuit originală bazată pe un principiu care rezolvă problema introducerii fără erori a corecției de fază, problemă fundamentală a realizării unui sincronizator de bit.

În lipsa tranzițiilor din fluxul de date recepționat, numărătorul binar de ieșire - de "n" biți - livrează tactul refăcut (de frecvență "f_c") cu un coeficiant de umplere de 50%. Vom presupune că tranzițiile sînt așteptate la frontul negativ al

tactului refăcut, în această situație frontul pozitiv fiind utilizat la eșantionarea datei recepționate; de asemenea, rezultă că dacă tranziția găsește tactul refăcut în starea logică "high", se impune corecția prin avansul fazei tactului generat, pentru a se realiza apropierea frontului negativ al acestuia de poziția tranziției, presupusă utilă (problema protecției la tranziții parazite va fi discutată ulterior); în mod corespunzător, corecția se efectuează cu întîrziere a fazei dacă tranziția găsește tactul refăcut în starea logică "low".

Metoda propusă ar putea fi denumită a ferestrelor de timp sincronizate, necesitînd generarea a două ferestre de timp cu duratele (T_1 și T_2) corelate între ele și cu perioadele impulsurilor de tact (de frecvențe "$2^{n+1} \times f_e$" și "$2^n \times f_e$"), după relația:

$$\frac{3}{4} \times T < T_1 < T_2 < T = \frac{1}{2^n \times f_e} .$$

Justificarea inegalităților de mai sus precum și a afirmației că fereastra de corecție T_2 trebuie sincronizată cu impulsurile de frecvență "$2^{n+1} \times f_e$" și "$2^n \times f_e$", începînd din starea 0 logic a acestora și încheindu-se tot în starea 0 logic a impulsurilor de frecvență "$2^n \times f_e$" - care sînt utilizate la obținerea tactului refăcut în afara ferestrei T_2 de corecție - se va face apelînd la fig. 2.

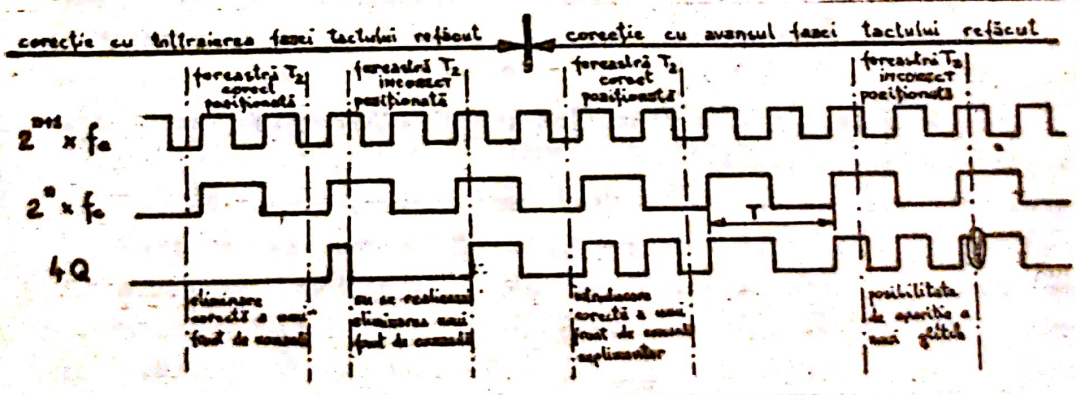


Fig. 2: Poziționarea necesară a ferestrei de corecție T_2.

Întîrzierea fazei tactului refăcut se realizează prin eliminarea în întregime a unui impuls pozitiv la intrarea de clock a numărătorului binar de "n" biți, ceea ce duce la suprimarea unui front pozitiv de comandă, în timp ce avansul fazei se realizează prin introducerea a două impulsuri de frecvență dublă în locul unui singur impuls pozitiv, asigurînd existența unui front pozitiv suplimentar de comandă a incrementării valorii binare a ieșirii acestuia numai în cazul poziționării corecte a ferestrei de corecție T_2. În consecință, rezultă că în ambele cazuri se introduce un salt de fază limitat ca valoare la $(360/2^n)$ grade.

Se poate face observația că, în vederea obținerii unei creșteri a vitezei de calare, fereastra de corecție T_2 poate fi prelungită cu un multiplu al perioadei T, pentru a ne asigura că sfîrșitul ei găsește tot în 0 logic impulsurile de frecvență "2^n x f_c"; această ultimă condiție permite, de exemplu, evitarea apariției unui scurt impuls "low" parazit la sfîrșitul ferestrei de corecție T_2 (această posibilitate fiind ilustrată în fig. 2, ultima fereastră de corecție). În cazul comutării (blocul 4, fig. 3) din starea 1 logic a impulsurilor de frecvență "2^{n+1} x f_c" tot în starea 1 logic a celor de frecvență "2^n x f_c", dacă 3Q ajunge în "low" în avans față de trecerea în "high" a lui $3\overline{Q}$. Oricum, după achiziția inițială viteza de calare trebuie micșorată, saltul de fază fix prescris la fiecare tranziție împiedicînd ieșirea rapidă din sincronism a tactului refăcut la o eventuală tranziție parazită.

Cele două ferestre de timp sînt generate neretriggerabil, din următoarele considerente:

- tranziții prea apropiate în timp (pe care le vom denumi, în continuare, tranziții adiacente), condiție care ne asigură că cel puțin una este parazită, vor genera o singură fereastra T_1;
- condițiile impuse asupra duratei ferestrelor ($T_2 > T_1 > (3/4)xT$), includ posibilitatea apariției a două impulsuri negative (pe durata T_1) la intrarea generatorului ferestrei T_2, care vor genera, însă, o singură fereastră de corecție T_2 în care se va opera asupra tactului refăcut.

3. Implementarea la nivel de circuit

Soluția propusă permite implementarea într-un circuit de o complexitate redusă, prezentat în fig. 3, care utilizează ca intrări fluxul de date primar demodulate și două tacte, cu coeficient de umplere 50%, de frecvențe "$2^{ord} \times f_c$" și "$2^n \times f_c$", generate extern. Circuitul de introducere sincronizată a corecției de fază este constituit din blocurile 1-4, blocurile suplimentar figurate (5-7) întregind o configurație de sincronizator de bit realizată în jurul circuitului propus.

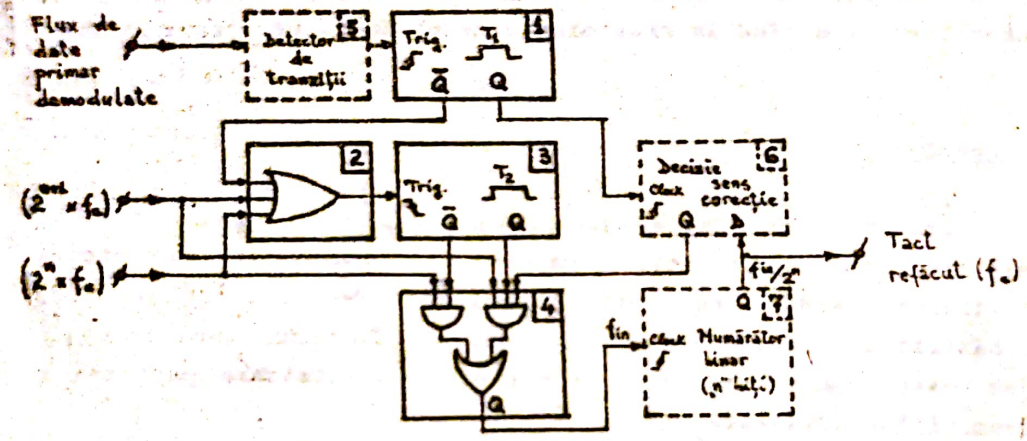


Fig. 3: Schema bloc a circuitului realizat.

Detectorul de tranziții (5), furnizează scurte impulsuri pozitive, cu durata mult mai mică decît T, la fiecare tranziție din fluxul de date recepționate, impulsuri al căror front pozitiv comandă generatorul neretriggerabil al ferestrei de durata T_1 (1), care astfel realizează filtrarea unor eventuale tranziții adiacente care ar apare pe durata T_1. Fereastra negativa (1 $\bar{Q}$), împreuna cu impulsurile de frecvență "$2^{ord} \times f_c$" și "$2^n \times f_c$", generează printr-o poartă SAU (2) impulsuri al căror front negativ declanșează generatorul neretriggerabil al ferestrei sincronizate de corecție T_2 (3). La frontul crescător al ferestrei pozitive T_1 (1 Q), circuitul de eșantionare și stabilire a sensului de corecție 6 memorează (6 Q) semialternanța tactului refăcut în care a apărut tranziția.

În lipsa ferestrei de corecție T_2, blocul de selecție și introducere a corecției (4), furnizează numărătorului divizor de frecvență prin "2" (7), impulsuri de frecvență "$2^n \times f_c$"; pe durata unei ferestre de corecție T_2, numărătorului îi este furnizat un tact (4 0) ce depinde de starea logică a lui 6 0 : impulsuri de frecvență "$2^{n+1} \times f_c$" pentru 1 logic și respectiv tact înghețat în 0 logic pentru 0 logic. Deci, prin modificarea lungimii semialternanțelor tactului refăcut cu cîte o perioadă T corespunzător fiecărei ferestre de corecție, se acționează în sensul apropierii frontului negativ al acestuia de tranzițiile din fluxul de date demodulate, scăderea factorului de divizare al numărătorului ducînd la creșterea vitezei de calare.

4. Concluzii

Circuitul ce implementează principiul propus prezintă avantajul unei simplități structurale deosebite, permițînd realizarea comodă a unui circuit integrat dedicat. Suplimentar, probabilitatea de ieșire din sincronism - în cazul unei recepții zgomotoase - este redusă prin filtrarea preliminară parțială a tranzițiilor adiacente.

BIBLIOGRAFIE:

1. BENNETT R. WILLIAM, DAVEY R. JAMES: "Data Transmission", Mc Graw-Hill 1965, parag. 14-1 - "Symbol timing for a synchronous binary system", pag. 260-262;
2. BRANA CODRUȚ, BRANA VALERIA: "Transmiterea informației numerice - vol. II", Editura Tehnică București 1970, Cap. 9 - "Sincronizarea de bit (sinfazarea)", pag. 79-83;
3. GREEN W. T. Jr, KEAN BILL: "Digital Phase-Locked Loops move into analog territory" - Electronic Design, March 31, 1982, pag. 95-100;
4. SMITH JACK: "Modern Communication Circuits", McGraw-Hill 1986, parag. 8.7 - "Digital Phase-Locked Loops", pag. 324-326.

Scanned with OKEN Scanner

METODA NECONVENTIONALA DE ACHIZITIONARE A INFORMATIEI DESPRE TEMPERATURA

As.ing. GALUPA NICOLAE
Catedra de Calculatoare
Facultatea de Automatica si Calculatoare
Institutul Politehnic Iasi

ing. MAYAYA KILOLE - KAKWAMA , doctorand
Facultatea de Automatica si Calculatoare
Institutul Politehnic Iasi

INTRODUCERE

Lucrarea de fata se incadreaza intr-o lucrare mai vasta axata pe executia de masuratori intr-un mediu industrial. Ceea ce vom prezenta se refera la achizitionarea informatiei despre temperatura in cazul unui mediu cu temperatura lent variabila. Elementul de noutate consta in faptul ca desi folosim ca traductor un termistor (element cu caracteristica puternic neliniara) nu vom folosi pentru conversie uzualele AD urmind apoi aplicarea unei tehnici de liniarizare , cit faptul ca prin insasi metoda folosita informatia rezultanta este caracterizata de liniaritate.

PREZENTAREA METODEI

Vom folosi ca traductor un termistor a carui caracteristica este descrisa de relatia :.

$$R(\theta)=Ae^{B/\theta}$$

unde A si B sint doua constante de material.

Executind o conversie in tensiune si o amplificare vom obtine o tensiune dependenta de temperatura dupa relatia :

DC

$$U(\theta)=De^{B/\theta}$$

Elementele de baza pe care ne vom baza pentru liniarizare vor fi tensiunile de incarcare si descarcare a unui condensator. Principial circuitul de masura este prezentat in fig. 1.

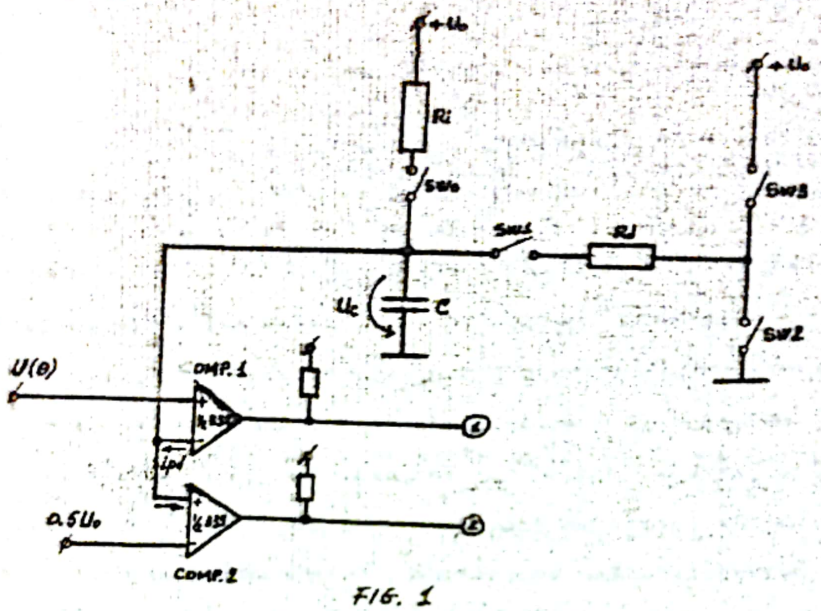


FIG. 1

Facem la inceput mentiunea ca sistemul este conectabil la un sistem cu microprocesor - deci iesirile notate cu 1 si 2 vor fi citite de catre acesta.

Presupunem initial ca Uc=0V si SW0=ON , SW1=SW2=SW3=OFF. Condensatorul se incarca pe circuitul +UO-Ri-C-GND. Legea de variatie a tensiunii la borne este descrisa de relatia :

$$Uc(t)=UO(1-e^{-t/RiC})$$

Un parametru caracteristic este produsul RiC. Il vom determina prin intermediul COMP2. Uc este tot timpul comparata cu 0.5UO -

Scanned with OKEN Scanner

ieșirea comparatorului fiind citită de microprocesor a.i. acesta din urmă contorizează timpul scurs din momentul Uc=0V pînă la momentul Uc=0.5U0. Notăm acest timp cu t1.

$$Uc(t1)=0.5U0 \Rightarrow 0.5U0=U0(1-e^{-t1/RiC}) \Rightarrow$$

$$\Rightarrow e^{t1/RiC}=2 \Rightarrow t1/RiC=\ln(2) \Rightarrow RiC=t1/\ln(2)$$

In continuare vom determina printr-o metodă similară produsul RdC. Pentru aceasta vom produce SW0=OFF , SW1=SW2=ON , SW3=OFF pentru un timp acoperitor a.i. să putem din nou face presupunerea că Uc=0V după care producem SW0=SW2=OFF , SW1=SW3=ON provocînd încărcarea condensatorului pe traseul +U0→Rd→C→GND. La fel se contorizează timpul în care se atinge 0.5U0 și-l notăm cu t2.

$$Uc(t2)=0.5U0 \Rightarrow 0.5U0=U0(1-e^{-t2/RdC}) \Rightarrow$$

$$\Rightarrow e^{t2/RdC}=2 \Rightarrow t2/RdC=\ln(2) \Rightarrow RdC=t2/\ln(2)$$

In continuare vom provoca la fel ca mai sus descărcarea condensatorului a.i. să putem presupune că Uc=0V. Din acest moment putem să începem a executa măsurători lucru pe care îl vom face în doi pași.

PAS 1 :SW0=ON , SW1=SW2=S3=SW3=OFF

Condensatorul începe se se încarce . Uc va atinge nivelul U0 după un timp infinit. Pentru a putea face măsurătoarea în timp finit alegem un factor k0<1 urmărind încărcarea condensatorului pînă la valoarea k0U0. Timpul necesar încărcării condensatorului va fi :

$$k0U0=U0(1-e^{-tc/RiC}) \Rightarrow tc=RiC\ln(1/(1-k0)) \Rightarrow$$

$$\Rightarrow tc=t1\ln(1/(1-k0))/\ln(2)$$

Deci microprocesorul care controlează interfața va menține

configuratia switch-urilor pentru timpul tc dupa care trecem la pasul 2 al procesului de masurare.

PAS 2 : SW1=SW2=OFF , SW0=SW3=ON

Condensatorul se va descarca legea de variatie a tensiunii la borne fiind descrisa de relatia :

$$Uc(t)=kOU0e^{-t/RdC}$$

Prin intermediul COMP1 comparam Uc(t) cu U(θx). Cit timp Uc(t)>U(θx) microsistemul contorizeaza timpul. Dupa tx avem :

$$Uc(tx)=U(\theta x) \Rightarrow kOU0e^{-tx/RdC}=De^{B/\theta x} \Rightarrow$$

$$\Rightarrow e^{(tx/RdC + B/\theta x)}=kOU0/D \Rightarrow$$

$$\Rightarrow \theta x=BRdC/(RdCln(kOU0/D)-tx) \Rightarrow$$

$$\Rightarrow \theta x=(Bt2/ln(2))/((Bt2/ln(2))ln(kOU0/D)-tx) \Rightarrow$$

$$\Rightarrow \theta x=Bt2/(Bt2ln(kOU0/D)-txln(2))$$

Presupunind ca microsistemul contorizeaza timpul dupa un ceas pilot de frecventa υ deci perioada τ vom putea exprima t1,t2,tx functie de perioada τ si trei numere N1,N2,Nx dupa cum urmeaza :

$$t1=N1\tau \ , \ t2=N2\tau \ , \ tx=Nx\tau$$

$$\theta x=BN2\tau/(BN2\tau ln(kOU0/D)-Nx\tau ln(2))$$

Notind Bln(kOU0/D)=α=const. si ln(2)=β=const. avem :

$$\theta x=BN2/(\alpha N2-\beta Nx)$$

unde N2 si Nx au fost anterior determinate cu ajutorul microsistemului.

Sa vedem care ar fi eroarea introdusa de comparatoare. Curentii lor de polarizare nu influenteaza curentii de incarcare

si descarcare ai condensatorului. Presupunem o eroare de comparare in tensiune de valoarea ΔU.

$$U(\theta)=\Delta U+Uc(tx) \Rightarrow$$

$$\Rightarrow De^{B/\theta}=\Delta U+kOUOe^{-tx/RdC} \Rightarrow$$

$$\Rightarrow B/\theta=\ln(\Delta U/D+(kOUO/D)e^{-tx/RdC}) \Rightarrow$$

$$\Rightarrow \theta=B/\ln(\Delta U/D+(kOUO/D)e^{-tx/RdC}) \Rightarrow$$

$$\Rightarrow \Delta\theta=(B/\ln(\Delta U/D+(kOUO/D)e^{-tx/RdC}))-(BRdC/(RdC\ln(kOUO/D)-tx)))$$

Studiind monotonia lui $\Delta\theta$ ne rezulta pentru o valoare minimala a erorii o valoare maximala pentru kO. Tinind cont de faptul ca kO intervine si in relatiile timpilor de incarcare/descarcare ai condensatorului C alegem kO=0,9.

Structura bloc a sistemului de masura este prezentata in fig.2.

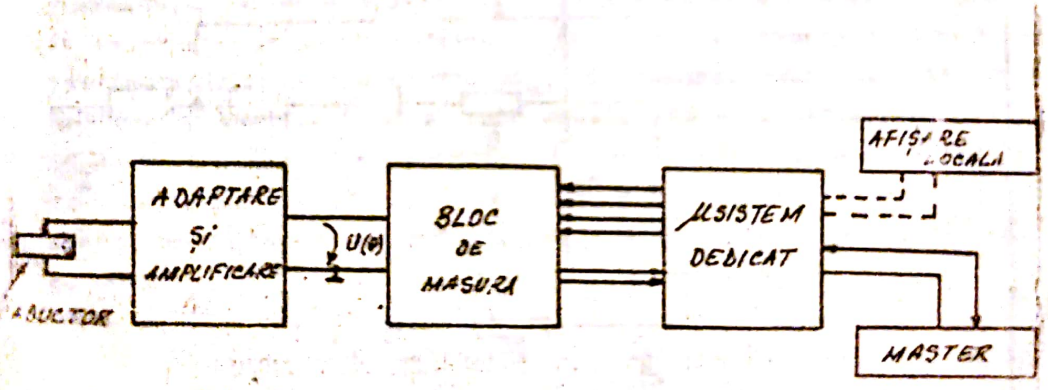
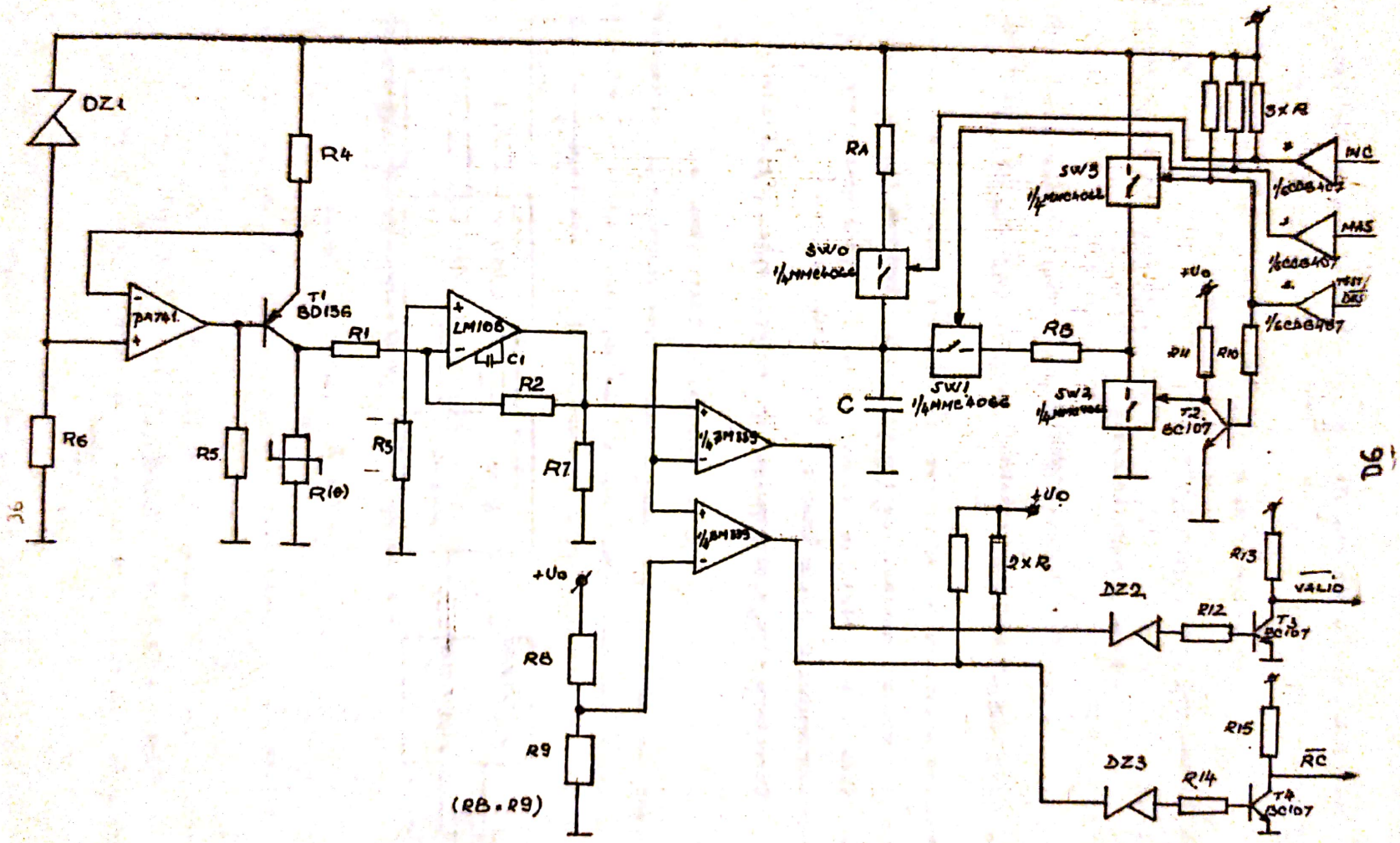


FIG. 2

Structura ce a fost implementata pentru testarea metodei este prezentata in fig. 3. Mentionam ca am folosit un calculator t


DZ1
R4
R6
R5
R(θ)
T1
BD136
R1
R3
LM108
C1
R2
R7
R8
R9
(R8 = R9)
+Uo
RA
SW0
1/4MMC4066
C
SW1
1/4MMC4066
R8
SW3
SW2
3xR
INC
MAS
R11
R10
T2
BC107
2xR
DZ2
R12
R13
VALID
T3
BC107
DZ3
R14
R15
RC
T4
BC107
D6

SINTEZA AUTOMATELOR SECVENTIALE UTILIZIND ARBORI MULTIVALENTI

Lélia Festila, Sorin Hintea
Institutul Politehnic Cluj- Napoca

REZUMAT

Se prezinta o metoda generalizata de sinteza circuitelor de tip automat secvential sincron. Procedeul utilizeaza functii logice multivalente reprezentate prin arbori de decizie logica.
Metoda se poate aplica atit pentru automatele secventiale binare cit si pentru automatele multivalente implementate cu multiplexoare (porti T) si bistabile D (multistabile B) si este direct aplicabila calculului automat.

1. Introducere

In cele ce urmeaza se prezinta o extindere a procedeului de sinteza a CLC bazat pe logica multivalenta [2] la sinteza automatelor secventiale sincrone, multivalente, inclusiv a celor binare. Generatorul unei stari are la baza un arbore de decizie fundamental, avind o configuratie simpla cu doua nivele. El poate fi restructurat pe orizontala si pe verticala pentru a fi adus la configuratia celulara de tip MX_{kr} [2] impusa. Notatiile utilizate, modificate partial fata de cele uzuale din descrierea diagramelor de decizie binara [1] sugereaza o legatura directa fata de diagramele de tranzitie aferente automatelor secventiale binare devenind astfel mai intuitive si mai usor de transpus in logica de calcul automat.

2. Descrierea automatului secvential in logica multivalenta

In cazul cel mai general automatul sincron este multivalent si are schema bloc data in figura 1. Semnalele o de iesire depind atit de semnalele i de intrare cit si de starea prezenta g. Schimbarea starii are loc in prezenta impulsului de tact CK. Intre doua impulsuri de tact starea este memorata de multistabilul B.

Proiectarea automatului de acest tip se refera de fapt la determinarea configuratiei blocului combinational, care stabileste relatia logica intre semnalul de iesire si marimile de intrare si de stare si care determina totodata starea viitoare. Deoarece acest circuit este de tip combinational, pentru sinteza lui se poate aplica procedeul bazat pe arborii multivalenti de decizie [2]. Ramine sa se indice modul de descriere logica a blocului combinational in acest cu acest procedeu.

Automatele de acest tip sint in general complexe, cu un mare numar de stari si semnale de intrare si sint descrise de functii logice de mari dimensiuni. Cu toate acestea, modelul propus in prezenta lucrare are o forma primara simpla, bidimensionala, datorata introducerii unei relatii logice intermediare multivalente.

Plecand de la observatia ca orice numarator de stare contine un numar de n stari distincte, iar tranzitia din fiecare stare, prezente se face spre cel mult n directii, in functie de unul sau mai multe semnale de comanda. De exemplu, numaratorul de stare din figura 2 are 7 stari distincte (n=7) si cel mult 3 directii d de tranzitie (n=3). Directia de tranzitie s-a considerat a fi o variabila n valenta. Fizic, ea reprezinta un semnal de intrare k valent (k ≤ n) sau un vector de semnale in cazul binar [2]

Numaratorul de stare poate fi descris de relatia implicita:

$$q^+ = f(q, d), \quad f: Q \times D \to P \qquad (1)$$

unde

$$d = g(q, i), \quad g: Q \times I \to D \qquad (2)$$

$$D = \{0,1 \ldots n-1\}, \quad Q = \{0,1,\ldots n-1\} \quad P = \{0,1 \ldots p-1\}$$

$$K = \{0,1 \ldots k-1\}, \quad I = \{i_1, i_2, \ldots, i_n\}$$

Relatia (1) se considera ca fiind fundamentala si i se poate atasa un arbore cu doua nivele. In acest sens exista doua variante, dupa cum terminatiile sint f(j,d); f(j,i) sau f(q,i), f(j,i) unde $j \in Q$, $i \in I$.

Relatia (2) se poate reprezenta printr-un arbore cu un nivel, avind ca terminatii semnalele i. Pentru descrierea completa a automatului se ataseaza si arborele corespunzator functiei de iesire $Y = h(q,i)$. In functie de tipul multiplexoarelor arborii se pot restructura, descompunindu-i pe orizontala si pe verticala

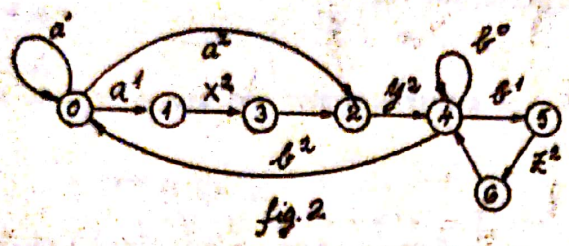


fig. 2

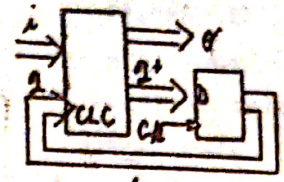


fig. 1

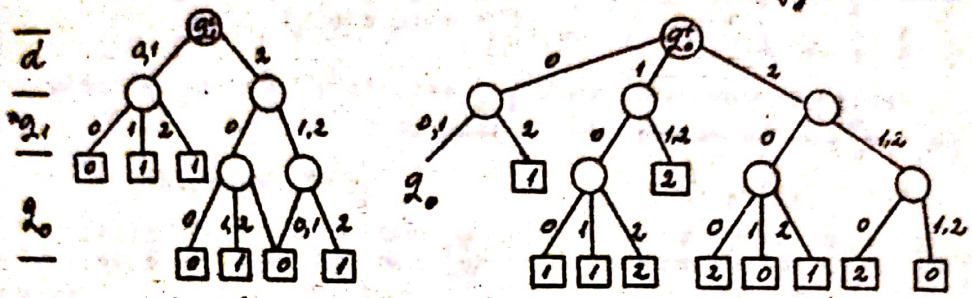


fig. 3

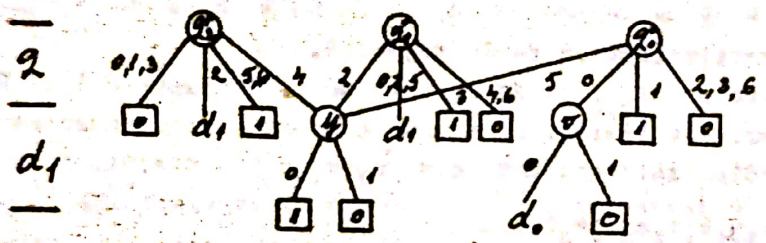


fig. 4

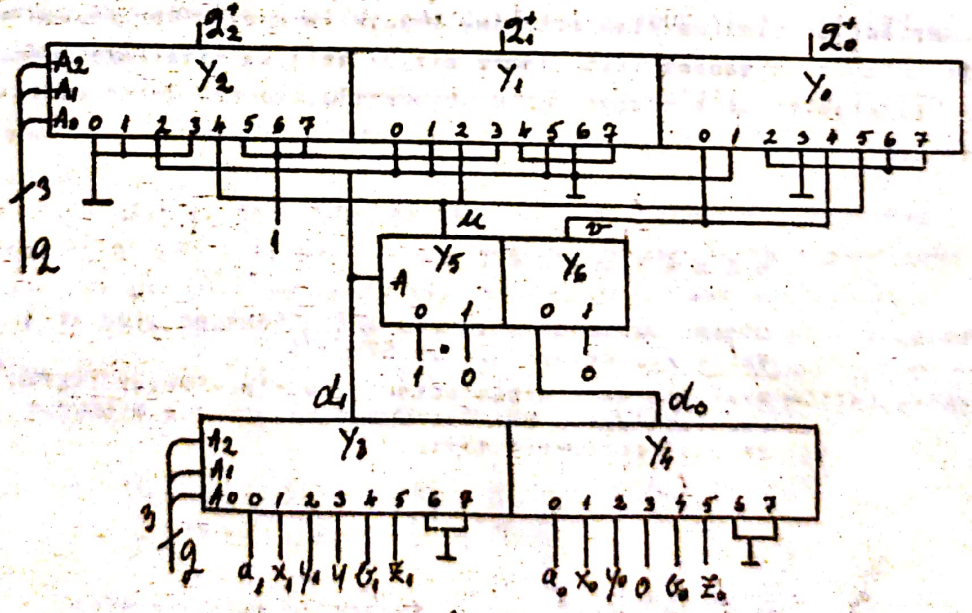


fig. 5

pentru a fi atunci în configuratia dorita

3. Exemple

Functiile (1) si (2) ale numaratorului de stare dat în figura 2 sînt indicate în tabelul 1

$q^+ = f(q,d)$

q \ d	0	1	2	3	4	5	6
0	0	1	2	-	4	5	4
1	1	1	2	-	5	5	-
2	2	3	4	2	0	6	-

$d = g(q,i)$

i \ d	0	1	2	3	4	5	6
	a	x	y	z	b	z	0

Tab.1

$i = [a\ x\ b\ z]$

Pentru implementare cu $MX_{3,3}$ functia q^+ devine vectoriala $q^+ = [q_1^+ \ q_0^+]$ si se reprezinta în tabelul 2.

$q_1^+ = f_1(q_1, q_0, d)$

q1 / q0 \ d	0 0	0 1	0 2	1 0	1 1	1 2	2 0
0	0 0	0 1	0 2	-	1 1	1 1	1
1	0 0	0 1	0 2	-	1 1	1 -	
2	0 1	1 0	0 0	2	-		

$q_0^+ = f_0(q_1, q_0, d)$

q1 / q0 \ d	0 0	0 1	0 2	1 0	1 1	1 2	2 0
0	0 1	1 2	2 -	1 1	1 1		
1	1 1	2 -	2 1	-			
2	2 0	1 2	0 0	-			

Tab.2

Arborele minimal multivalent se da în figura 3.

b) Varianta minimala cu $MX_{2,8}$ si $MX_{2,2}$ este data în figurile 4 si 5 care reprezinta arborele minimal si schema aferenta, corespunzatoare CLC a generatorului unei stari : $q = [q_2\ q_1\ q_0]$, $d = [d_1\ d_2]$

$i = [i_1, i_0] = [a_1 x_1 y_1 b_1 z_1 a_0 x_0 y_0 b_0 z_0]$

4.- Concluzii

Metoda de sinteza elaborata înglobeaza în mod unitar si sistematic atît automatele secventiale binare cît si cele multivalente, indiferent de complexitatea grafului lor de tranzitie. Ea are certe avantaje în calculul automat.

B I B L I O G R A F I E

1 Malden, S.A - Binary Decision Diagrams IEEE TRans.on Computers, Vol C-27, nr.6. 1978 pp 509-516

2 Festila,Lelia- Procedeu de sinteza automata a circuitelor logice combinationale multivalente.Comunicarea Simp. IP Iasi octombrie 1991.

PROCEDEU DE SINTEZA AUTOMATA A CIRCUITELOR LOGICE COMBINATIONALE MULTIVALENTE

Lelia Festila - Institutul Politehnic
Cluj - Napoca

REZUMAT

Metoda propusa se refera la sinteza circuitelor logice combinationale multivalente utilizind celule functionale de tip multiplexor. Se indica modul de realizare a arborelui de decizie aferent functiei logice date si tipului de celule dorit. Utilizarea arborelui de decizie permite automatizarea procedeului de sinteza si de cautare a unor solutii minimale.

1. Introducere

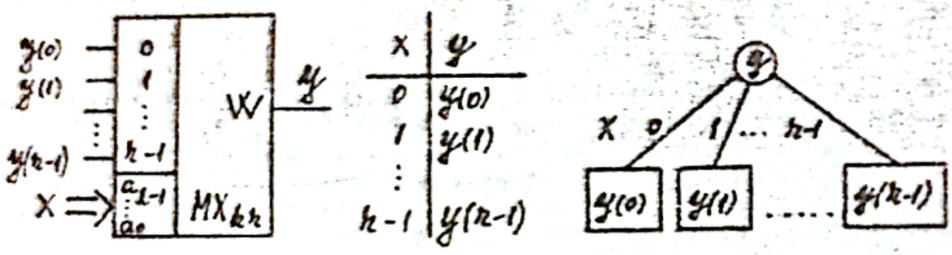
In ultimii ani s-au extins studiile si realizarile legate de logica multivalenta si circuitele larg integrate bazate pe aceasta logica [1] . Acestea sint circuite care prelucreaza semnale cu mai multe nivele logice. Utilizarea lor permite micsorarea ariei de siliciu si reduce numarul conexiunilor interne si a liniilor de magistrala.

In cele ce urmeaza am utilizat logica multivalenta pentru a dezvolta un procedeu generalizat de proiectare a CLC, procedeu care se aplica atit la sinteza circuitelor logice multivalente cit si a celor binare, care apar ca si cazuri particulare ale primei categorii. Arborele de decizie atasat circuitului cautat este compus din celule functionale de tip multiplexor. Configuratia bazata pe celule functionale este proprie circuitelor larg integrate. Arborele de decizie poate fi creat si minimizat automat permitind analiza mai multor variante pentru cautarea solutiilor optime.

2. Multiplexorul k-valent, celula fundamentala a arborelui de decizie logica.

Multiplexorul k-valent cu r intrari MX_{kr} (fig.1) prezinta urmatoarele caracteristici:

- are r intrări de date și o ieșire;
- semnalele prelucrate de MX sînt k-valente (cînd k>2) circuitul se mai numește T [1] ;
- oricare din semnalele i de intrare poate fi transmis la ieșire selectîndu-l cu un semnal x de adresă, sau de selecție;
- semnalul x poate fi singular sau compus ; cînd r>k, adresa se formează pe mai multe intrări, prin transpunerea în baza k a fiecărei valori zecimale x^i a semnalului de selecție.



a) b) c)

fig.1.

$$x = \overline{a_{l-1}, a_{l-2}, \ldots, a_0} = \sum a_i k^i \qquad (1)$$

astfel încît $x^i = \overline{a^i_{l-1}, \ldots, a^i_0}$; $a_i \in \{0, \ldots, k-1\}$,

$x^i \in \{0, \ldots, r-1\}$, $l = \log_k r$

Se poate observa că orice multiplexor MX_{kr} realizează funcția logică multivalentă.

$$y = f(x),\ f: R \to k,\ R = \{0,1,\ldots r-1\},\ K = \{0,1,\ldots k-1\} \qquad (2)$$

dată în figura 1,b.

Valoarea funcției y se obține la ieșirea MX. Variabila x are semnificația fizică în cazul în care r≤k reprezentînd semnalul de adresă. Cînd r>k, r este dodul zecimal al adresei multiplexorului.

Funcția (2) se poate reprezenta prin arborele de decizie din figura 1,c în care în nodurile terminale apar valorile y(i) ale funcției corespunzătoare valorilor $x_i \in R$ ale variabilei x, valori indicate lîngă ramurile arborelui radial.

b) orice multiplexor poate realiza funcția logică de q variabile:

$$y = g(a_{p-1}, a_{p-2}, \ldots a_0),\ g : k^p \to K \quad p<l = \log_k r \qquad (3)$$

a_j, $j=0,\ldots l-1$ fiind semnale singulare aplicate la intrarile de adresa iar y este iesirea multiplexorului.

3. Implementarea functiilor logice multivalente cu Mx_{kr}

Se pune problema implementarii functiei logice multivalente:

$$y = f(x),\ f:H \to H,\quad H=0,1\ldots n-1,\quad H=0,1\ldots n-1 \qquad (4)$$

Se va utiliza un multiplexor k valent de r căi MX_{kr}

a) Cazul $n \leq r$, $n \leq k$

Implementarea este directa, conform figurii 1 si cu observatiile facute la paragraful 2 privitor la formarea adresei (variabila x este decodificata in baza k).

b) Cazul $n \leq r$ $n>k$

Multimea valorilor functiei este mai mare decit numarul nivelelor logice ale semnalelor prelucrate de Mx_{kr}. In acest caz valorile numerice ale functiei se decodifica in baza k, iar functia y devine vectoriala

$$y = [y_{l-1}, y_{l-2}, \ldots y_0] = [y_i]\ l \qquad (5)$$

$$y_j = f_j(x),\ f_j : H \to K,\quad l = (\ln_k n) + 1, y(i) =$$

$$y_{l-1}^{(i)}\ y_{l-2}^{(i)} \ldots y_{ok}^{i},\ i \in H$$

Arborele din fägüra 1,b de descompune pe orizontala in l arbori in paralel, aflati pe acelasi nivel. Fiecare graf $j = 0,\ldots l-1$ avind ca terminali $y_j(i)$, $i = 0\ldots n-1$.

c) Cazul $n>r$, $n \leq k$

Cind multimea valorilor variabilei este mai mare decit numarul de intrari ale multiplexorului, fiecare element din multimea H se va reprezenta in baza r, rezultind variabila vectoriala.

$$x = [x_{l-1}, x_{l-2}, \ldots x_0] = x_j = [x_j]_l\ l : j=0,\ldots l-1\ l=\log_r n+1 \qquad (6)$$

astfel incit pentru fiecare valoare $x^i \in H$, $x^i = [x^i_{l-1}, \ldots x^i_0]$.

funcția logică devine

$$y = g(x_{i-1}, x_{i-2}, \ldots, x_0),\ g : B^i \to B \qquad (7)$$

Iar arborele de decizie (figura 2) va descompune pe verticale în i nivele, fiecare conținând un număr de noduri funcționale de tip MX_{kr}.

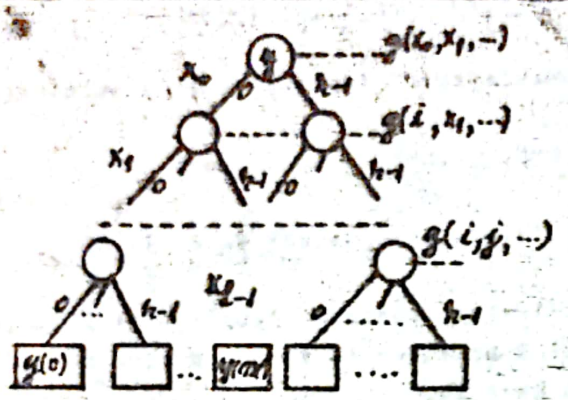


fig. 2.

Ca urmare pentru implementarea funcției se vor folosi multiplexoare MX_{kr} conectate în cascadă. Arborele nu este unic ci depinde de ordinea de explicitare a variabilelor pe nivele 0,...i-1, ca urmare vor exista i noduri de implementare posibile.

d) Cazul n >r, e>r

Se realizează descompunerea pe orizontală și pe verticală a arborelui aferent funcției date, după procedeele descrise mai sus.

4. Etapele proiectării CLC cu multiplexoare.

În capitolul 3 s-a arătat că orice funcție logică y=f(x) de forma (4) poate fi implementată cu MX_{kr} . Afirmația se extinde și la funcții de mai multe variabile,, caz în care x este un vector de forma (1) sau (6). Etapele proiectării CLC cu MX_{kr} vor fi următoarele:

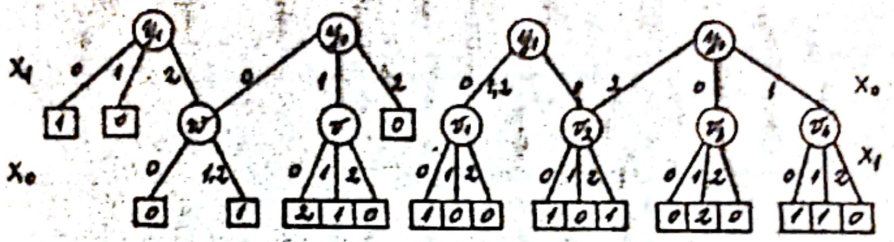
- se definește funcția sau funcțiile cerute,
- funcția logică se aduce la forma (4) (5) sau (7),
- se construiește arborele de decizie aferent acestor forme,
- se reduce arborele prin eliminarea buclelor redondante.

Condițiile de redondanță sînt îndeplinite dacă:

- valorile terminale ale unui nod funcțional sînt egale între ele.
- două celule funcționale plasate pe aceleași nivel au aceleași

configuratie.

Procedeele de reducere sînt iterative şi ierarhizate, ca urmare se pot automatiza. Algoritmul de definire a arborelui permite permutarea variabilelor sale pe diverse nivele pentru compararea variantelor rezultate în urma minimizării.



a) fig.3 b)

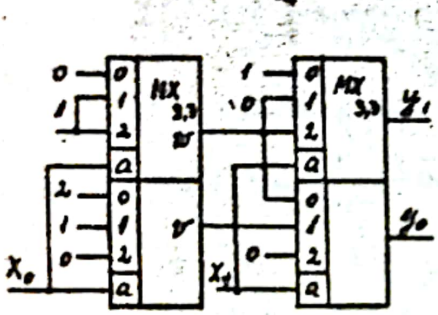


fig.4

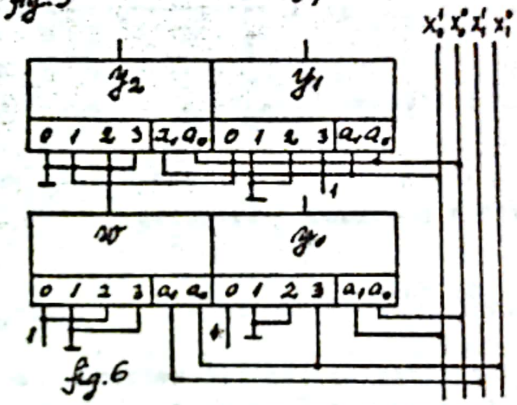


fig.6

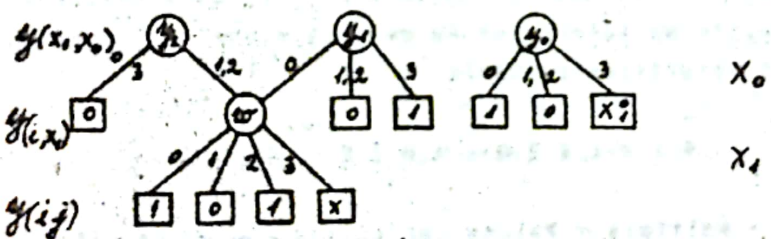


fig.5

5.- Exemple

Se va implementa functia logica y = f(x), f: N → N, N = [0,...,8]; [N = 0,...,4] data in tabelul 1. Se vor utiliza

a) $MX_{5,2S}$; b) $MX_{3,3}$; c) $MX_{2,4}$

x	0	1	2	3	4	5	6	7	8
y	3	4	4	2	1	0	0	3	3

Tabelul 1

a) Adresa se va exprima prin 2 semnale $[a_1\ a_0] = [00, 01, 02, 03, 04, 10, 11, 12, 13]$ iar valorile functiei se aplica direct la primele 9 intrari ale multiplexorului.

b) Arborele functiei va avea doua nivele si doua ramuri in paralel, corespunzatoare functiei date in tabelul 2.

x_1	0	0	0	1	1	1	2	2	2
x_0	0	1	2	0	1	2	0	1	2
y_1	1	1	1	0	0	0	0	1	1
y_0	0	1	1	2	1	0	0	0	0

Tabelul 2

x_1	0	0	0	0	1	1	1	1	2
x_0	0	1	2	3	0	1	2	3	0
y_2	0	1	1	0	0	0	0	0	0
y_1	1	0	0	1	0	0	0	1	1
y_0	1	0	0	0	1	0	0	1	1

Tabelul 3

In figurile 5 si 6 se indica varianta minimala a arborelui, respectiv a schemei rezultate.

5. Concluzii.

Utilizarea logicii multiple largeste aria posibilitatilor de proiectare a CLC cu MX. Procedeul prezentat permite efectuarea unui studiu sistematic pe calculator in vederea stabilirii configuratiei si alegerii solutiilor optimale.

B I B L I O G R A F I E

1. Hurst, S.L. - Multiple - Valued Logic. Its Status and Its Future. IEEE Trans on Computers, C.33 nr.12 1984.

2. Karpovski,M - Multilevel logical networks, IEEE Trans. on Computers, C-36, 2, 1987.

SISTEME SECVENTIALE-ANALIZA/SIMULARE

FLORIN HOZA

1. Introducere

Lucrarea respectiva prezinta o metoda de analiza a sistemelor secventiale care poate fi folosita fie pentru sisteme secventiale clasice fie pentru automate cablate.

Metoda se preteaza la implementarea pe microcalculatoare si consta in stabilirea unei matrici care evidentiaza evolutia starilor sistemului.

Este necesar sa se prezinte citeva notiuni utile:

a). fie conjunctia notata Q de "n" variabile $x_{n-1},.....x_o$

- se defineste exponentul formal - z - pentru o variabila binara

(1) $$z = \begin{cases} 0 - \text{daca variabila binara x apare complementata } (\bar{x}) \\ 1 - \text{daca variabila binara x apare necomplementata } (x) \end{cases}$$

- astfel conjunctia poate fi scrisa

(2) $$Q = \sum x_i^{z_i}$$

b). echivalentul zecimal reprezinta valoarea zecimala a unui numar reprezentat binar (ex.echivalentul zecimal pentru (1011) este 11))

- se completeaza Q cu un indice k care reprezinta echivalentul zecimal al numarului binar format din exponentii formali z_i

(ex.pentru $Q = \bar{x}_2 x_1 \bar{x}_o \Rightarrow Q = x_2^o x_1^1 x_o^o \Rightarrow k=(010)=2 \rightarrow Q_2 = x_2^o x_1^1 x_2^o$

ceea ce permite identificarea fiecarei conjunctii prin indicele atasat.

2. Analiza sistemelor secventiale

Fie un sistem secvential cu "p" stari si "q" intrari,descris de n ecuatii de stare

(3) $$y_i = f_i(x_{q-1},.....x_o,\ y_{n-1},.....y_o) \qquad i = \overline{o,n-1}$$

Nota: Textele din chenar reprezinta indicatii si exemple

unde f_i reprezintă o sumă de conjuncţii.

Dar o conjuncţie din care lipseşte cel puţin o variabilă (forma minimizată) poate fi scrisă ca o sumă de conjuncţii elementare (ex. fie $n = 2$ şi $Q = \bar{y}_2 y_0 \Rightarrow Q = \bar{y}_2\bar{y}_1 y_0 + \bar{y}_2 y_1 y_0$).

Astfel o ecuaţie de stare poate fi scrisă:

$$(4)\quad Y_i = \sum_j \left(g_{ij}(x_{q-1},\ldots\ldots,x_0).\left(\sum_\ell Q_\ell\right)\right)$$

(ex: dacă $y_2 = xy_1\bar{y}_0 + \bar{x}y_2y_1 + y_2\bar{y}_1y_0$ atunci $y_2 = x(\bar{y}_2y_1\bar{y}_0 + y_2y_1\bar{y}_0) + \bar{x}(y_2y_1\bar{y}_0 + y_2y_1y_0) + y_2\bar{y}_1y_0$ unde $g_{21} = x$, $g_{22} = \bar{x}$, $g_{23} = 1$)

In expresia (4) o conjuncţie Q_k poate să apară de mai multe ori însoţită de funcţii g_{ij} diferite. Prin gruparea permanenţilor Q_k se obţine pentru y_j expresia generală:

$$(5)\quad Y_i = \sum_\ell n_{i\ell}\, Q_\ell$$

Dacă se consideră sistemul cu următoarele ecuaţii:

$$(5.1)\begin{cases} Y_2 = \bar{y}_2\bar{y}_1 x + \bar{y}_2\bar{y}_1 y_0 + y_2y_1\bar{x} + \bar{y}_1\bar{y}_0 x \\ Y_1 = \bar{y}_2y_1\bar{y}_0 + \bar{y}_2y_0\bar{x} \\ Y_0 = \bar{y}_2y_0 + y_1\bar{y}_0\bar{x} + \bar{y}_1\bar{y}_0\bar{x} \end{cases}$$

atunci:

$$(5.2)\begin{cases} Y_2 = y_2^0y_1^0y_0^0x + y_2^0y_1^1y_0^0x + y_2^0y_1^1y_0^1 + y_2^1y_1^0y_0^0x^0 + y_2^1y_1^0y_0^1x^0 + \\ \quad + y_2^1y_1^1y_0^0x^0 + y_2^1y_1^1y_0^1x^0 \\ Y_1 = y_2^0y_1^0y_0^1x^0 + y_2^0y_1^1y_0^0 + y_2^0y_1^1y_0^1x^0 \\ Y_0 = y_2^0y_1^0y_0^0 + y_2^0y_1^1y_0^0 + y_2^1y_1^0y_0^0x^1 + y_2^1y_1^1y_0^0x^0 \end{cases}$$

sau:

$$(5.3)\begin{cases} Y_2 = Q_0 + xQ_1 + Q_3 + xQ_4 + \bar{x}Q_5 + \bar{x}Q_6 + \bar{x}Q_7 \\ Y_1 = \bar{x}Q_1 + Q_2 + \bar{x}Q_3 \\ Y_0 = Q_0 + Q_2 + xQ_4 + \bar{x}Q_6 \end{cases}$$

Se trece uşor de la ecuaţiile (5) la o ecuaţie matricială

$$(6)\quad \begin{bmatrix} Y_{n-1} \\ Y_{n-2} \\ \vdots \\ Y_1 \\ Y_0 \end{bmatrix} = \begin{bmatrix} k_{n-1,0} & k_{n-1,1} \cdots\cdots k_{n-1,2^n-1} \\ k_{n-2,0} & k_{n-2,1} \cdots\cdots k_{n-2,2^n-1} \\ \vdots & \vdots \\ k_{1,0} & k_{1,1} \cdots\cdots k_{1,2^n-1} \\ k_{0,0} & k_{0,1} \cdots\cdots k_{0,2^n-1} \end{bmatrix} \cdot \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ q_{2^n-2} \\ q_{2^n-1} \end{bmatrix}$$ sau $Y = K \cdot Q$

unde:

$$K_{ij} = \begin{cases} h_{ij} & \text{dacă } q_j \subset Y_i \\ 0 & \text{dacă } q_j \not\subset Y_i \end{cases}$$

Matricea K pune în evidență succesiunea stărilor; starea curentă este dată de indicele unei coloane, iar starea următoare este dată de echivalentul zecimal al numărului binar format din elementele coloanei (ex: din starea 0 (S_0) sistemul va evalua spre starea S_m, unde: $m = (k_{n-1,0}\ k_{n-2,0} \cdots\cdots k_{1,0}\ k_{0,0})$.

Pentru cazul particular caracterizat de ecuațiile (5.7) se obține ecuația matricială:

$$(6.1)\quad \begin{bmatrix} Y_2 \\ Y_1 \\ Y_0 \end{bmatrix} = \begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{bmatrix} x & x & 0 & 1 & x & \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & 1 & \bar{x} & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & x & 0 & \bar{x} & 0 \end{bmatrix} \end{array} \cdot \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ q_7 \end{bmatrix}$$

$$x = 0 \rightarrow \begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \quad ; \quad x = 1 \rightarrow \begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Obs: deasupra primei linii din matrice s-au trecut stările curente.

Din starea 0 - S_0, sistemul va evalua spre starea dată de (x 0 1), iar din starea 3 - S_3 spre starea dată de (1 x 0); pentru $x = 0$: $S_0 \rightarrow S_1$ și $S_3 \rightarrow S_6$, iar pentru $x = 1$: $S_0 \rightarrow S_5$ S_1 $S_3 \rightarrow S_4$

Pe baza matricei K se poate stabili graful sistemului (fig.1).

Metoda se aplică similar și pentru sisteme secvențiale cu mai multe intrări, situație în care elementele k_{ij} vor fi funcții de

mai multe variabile:

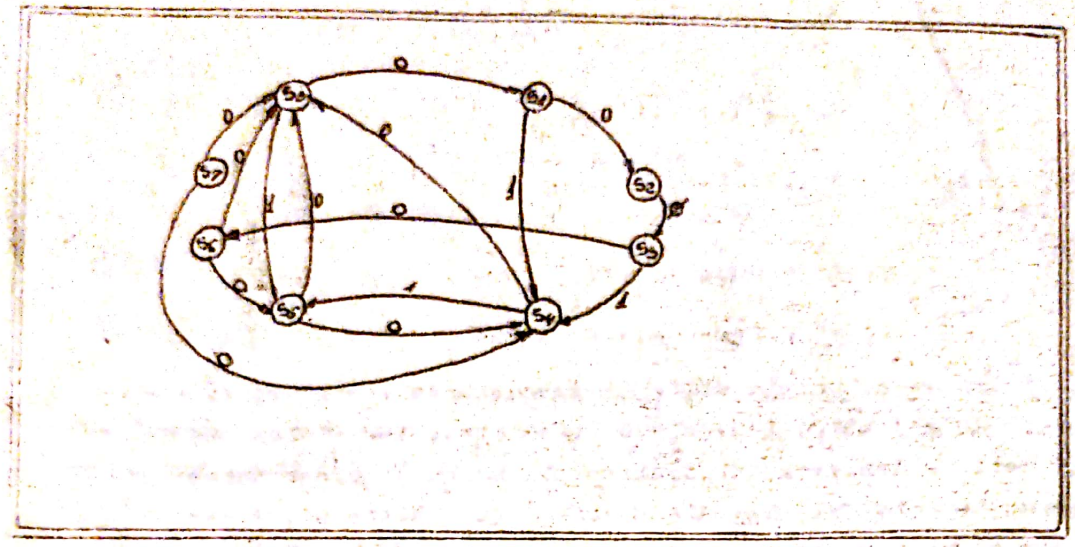


Fig.1.

3. Implementarea algoritmului

Fie un sistem secvențial caracterizat de n ecuații de stare (n variabile de stare) asupra cărora se aplică metoda prezentată

Implementarea necesită rezervarea unui cuvînt pentru fiecare conjuncție din cei (2^n-1) conjuncții elementare care conțin exponenții formali ai conjuncțiilor, începînd cu Q_0 și terminînd cu Q_{2^n-1}. Fiecărei conjuncție i se aplică o variabilă P_k (inițial 0), care reprezintă de fapt termenul k_{ij}. La introducerea unei conjuncții din expresia variabilei de stare Y_{ij} se generează două cuvinte de n biți: primul are poziționați biții corespunzători variabilelor de stare care se găsesc în conjuncție, iar al doilea cuvînt exponenții formali ai variabilelor de stare. Aceste două cuvinte sînt folosite pentru toate conjuncțiile din expresiile variabilelor de stare. Se înmulțește logic primul cuvînt cu cuvintele corespunzătoare celor 2^n-1 conjuncții elementare și se compară rezultatul cu al doilea octet (referința (2)); în caz de identitate se sumează P_i cu funcția &, care însoțește conjuncția (rel. (4)); dacă $P_j = 1$ se consideră $P_j = 1$ și nu se mai execută alte sumări.

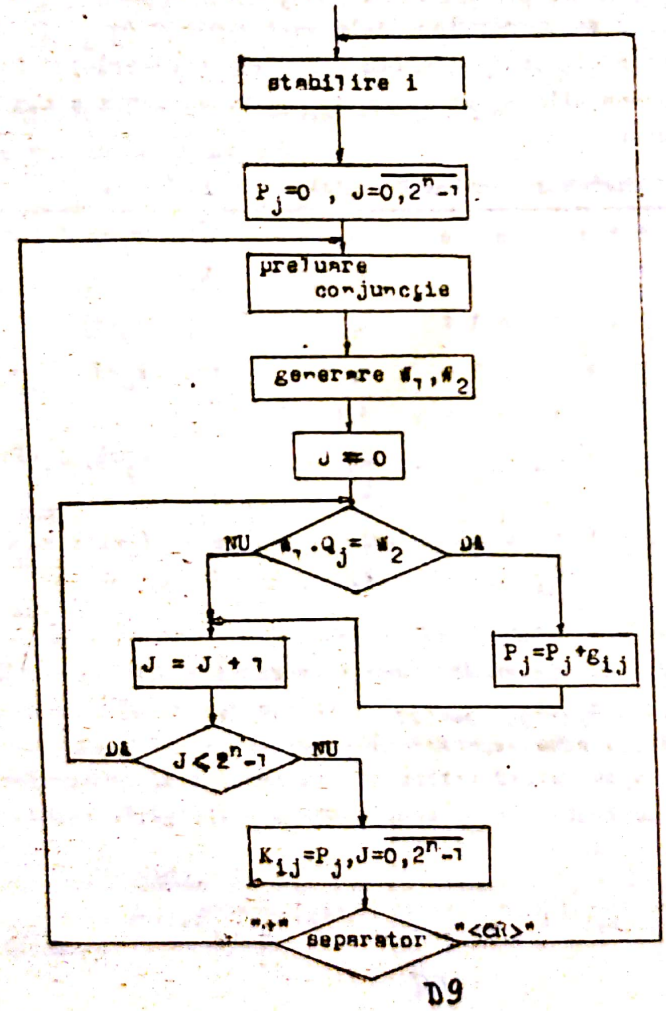
Operațiile se repetă pentru fiecare conjuncție din expresia lui Y_i, după introducerea acestuia, încît după introducerea ecuației linia corespunzătoare din matrice este completă și se

poate prelua din zona P; acest lucru permite ca zonele de memorie de lucru să fie utilizate pentru următoarea ecuație

Dacă vectorul de intrare conține mai multe variabile în introducerea acestora se stabilește ușor dacă un element al termenilor k_{ij} este 1 prin compararea exponenților formali cu valorile particulare (fie $Q_k = \prod x_j^{z_j}$; atunci $Q_k = 1 \Leftrightarrow x_j = z_j$)

Se consideră ca separatori caracterele funcționale :" = ", " + ", " < CR > ".

Organigrama de principiu este prezentată în fig. 2.



D9

Se exemplifică metoda pentru sistemul dat de ecuațiile (5.7)

- în memorie există tabela:

(6.2)

0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

- după introducerea primei conjuncții separată de operatorul + , (= $x(y_2y_1$) se construiesc cele două cuvinte de conținut (1,1,0) și (0,0,0); prin algoritmul prezentat referința (2); se identifică conjuncțiile Q_0 și Q_1 - $k_{2,0} = x$, $k_{2,1} = x$ ș.a.m.d. conform tabelului:

conjuncție Y_2	cuvînt 1	cuvînt 2	$s_1 \cdot Q_j = s_2$	I	$k_{i,j}$
$x(\bar{y}_2\bar{y}_1)$	1 1 0	0 0 0	Q_0	0	$P_0=x$, $P_1 = x$
			Q_1	1	
$\bar{y}_2y_1y_0$	1 1 1	0 1 1	Q_3	3	$P_3 = 1$
$\bar{x}(y_2y_1)$	1 1 0	1 1 0	Q_6	6	$P_6=\bar{x}$, $P_7 = \bar{x}$
			Q_7	7	
$\bar{x}(y_2y_1)$	1 0 1	1 0 1	Q_5	5	$P_5=\bar{x}$, $P_7=\bar{x}+\bar{x}=\bar{x}$
			Q_7	7	
$x(\bar{y}_1\bar{y}_0)$	0 1 1	0 0 0	Q_0	0	$P_0=x+x=x$, $P_4=x$
			Q_4	4	

Tab. 1.

- elementele $k_{2,j}$ se preiau din zona P (variabilele $P_0,\ldots,P_7$)
  $(k_{20},k_{21},k_{23},k_{24},k_{25},k_{26},k_{27}) = (x,x,0,1,x,\bar{x},\bar{x},\bar{x})$
- ultimă conjuncție este separată de caracterul <CR>
- Se pot genera apoi cele 2 valori ale variabilei x, obținîndu-se cele 2 matrici derivate din K, deci toate stările și tranzițiile sistemului secvențial.

Pentru mai multe variabile de intrare, sistemul poate cere o anumită configurație a vectorului de intrare, determinînd evoluția stărilor.

Se exemplifică metoda pentru sistemul dat de ecuațiile (5.7)

- în memorie există tabela:

(6.2)

0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

- după introducerea primei conjuncții separată de operatorul ' , ' ($x(y_2y_1$) se construiesc cele două cuvinte de conținut (1,1,0) și (0,0,0); prin algoritmul prezentat referința (2)$_3$ se identifică conjuncțiile Q_0 și Q_1 - $k_{2,0} = x$, $k_{2,1} = x$ ș.a.m.d. conform tabelului:

conjuncție I_2	cuvînt 1	cuvînt 2	$w_j \cdot Q_j = w_2$	I	$k_{i,j}$
$x(\bar{y}_2\bar{y}_1)$	1 1 0	0 0 0	Q_0	0	$P_0=x$, $P_1 = x$
			Q_1	1	
$\bar{y}_2y_1y_0$	1 1 1	0 1 1	Q_3	3	$P_3 =1$
$\bar{x}(y_2y_1)$	1 1 0	1 1 0	Q_6	6	$P_6=\bar{x}$, $P_7 = \bar{x}$
			Q_7	7	
$\bar{x}(y_2y_1)$	1 0 1	1 0 1	Q_5	5	$P_5=\bar{x}$, $P_7=\bar{x}+\bar{x}=\bar{x}$
			Q_7	7	
$(\bar{y}_1\bar{y}_0)$	0 1 1	0 0 0	Q_0	0	$P_0=x+x=x$, $P_4=x$
			Q_4	4	

Tab. 7

- elementele $k_{2,j}$ se preiau din zona P (variabilele $P_0, \ldots, P_7$)
  $(k_{20}, k_{21}, k_{23}, k_{24}, k_{25}, k_{26}, k_{27}) = (x, x, 0, 1, x, \bar{x}, \bar{x}, \bar{x})$
- ultima conjuncție este separată de caracterul <CR>
- Se pot genera apoi cele 2 valori ale variabilei x, obținîndu-se cele 2 matrici derivate din A, deci toate stările și tranzițiile sistemului secvențial.

Pentru mai multe variabile de intrare, sistemul poate cere o anumită configurație a vectorului de intrare, determinînd evoluția stărilor.

Similar se poate proceda pentru vectorul de ieşire Z. Presupunem că sistemul secvenţial are "u" ieşiri: Z_{u-1}, Z_0. Similar ca pentru variabile de stare, folosind aceeaşi metodă, se ajunge la relaţia (7).

$$(7)\quad \begin{bmatrix} Z_{u-1} \\ Z_{u-2} \\ \vdots \\ Z_1 \\ Z_0 \end{bmatrix} = \begin{bmatrix} m_{u-1,0} \cdots\cdots\cdots\cdots m_{u-1,2^n-1} \\ m_{u-2,0} \cdots\cdots\cdots\cdots m_{u-2,2^n-1} \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ m_{1,0} \cdots\cdots\cdots\cdots m_{1,2^n-1} \\ m_{0,0} \cdots\cdots\cdots\cdots m_{0,2^n-1} \end{bmatrix} \cdot \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ \vdots \\ q_{2^n-1} \end{bmatrix}$$

unde elementele de pe coloana "i" reprezintă valorile variabilelor de ieşire corespunzătoare stării "i".

După introducerea ecuaţiilor de ieşire, programul afişează stările codificate prin numărul lor şi valorile corespunzătoare ale variabilelor Z.

Ca exemplu, considerând că pentru sistemul descris de ecuaţiile (5.1) ieşirile sînt:

$$(7.1)\quad \begin{cases} Z_1 = y_2 y_0 + y_2 y_0 \\ Z_0 = y_0 \end{cases}$$

se obţine:

$$(7.2)\quad \begin{bmatrix} Z_1 \\ Z_0 \end{bmatrix} = \begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array} \cdot \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ \vdots \\ q_7 \end{bmatrix}$$

- pentru stare 3 - ieşirea va fi $(Z_1, Z_0) = (1,0)$

Prelucrarea ecuaţiilor de ieşire are loc după aceeaşi metodă ca la ecuaţiile de stare.

Ca şi în referinţa (2) sistemul poate fi prevăzut cu o interfaţă paralelă, devenind un sistem - expert de simulare a sistemelor secvenţiale.

4. Concluzii

Faţă de alte metode - referinţa (2), această metodă prezintă următoarele avantaje:

- o zona relativ mica de memorie - cite un cuvint pentru fiecare conjunctia Q_k din cele 2^n-1 conjunctii elementare:
- zona de memorie corespunzatoare conjunctiilor dintr-o ecuatie este comuna, in sensul ca este folosita pentru toate ecuatiile,deoarece dupa introducerea unei ecuatii se poate determina complet linia corespunzatoare din matricea K (eventual pe durata introducerii urmatoarei linii)
- timp de executie foarte mic, din punct de vedere utilizator, deoarece o linie- se calculeaza imediat de ecuatia de stare a fost introdusa.
- raspuns rapid dupa introducerea unui vector de intrare deoarece expresiile elementelor k_{ij} au fost stabilite anterior

Bibliografie

1. Al.Valachi, M.Birsan - Tehnici numerice si Automate, Ed.Junimea , 1987
2. I.Dancea - Dynamically changing the Logical Behaviar of a Microcomputer Interface
3. Vangalur S.Alagar - Fundamentals of computing, Prentice Hall, 1989
4. H.Taub - Digital circuits and microprocessors McGraw-Hill, 1985
5. C.R.Clare - Designing logic systems using states machines McGraw-Hill, 1972
6. G.Russell - Computer aided tools for VLSI system desing, Peter Peregrinusltd 1987

UTILIZARE INEDITA A TIMERULUI I8253 IN COMUNICATII SERIALE

prof. dr. ing. Al. Valachi
I.P. Iasi
asist. ing. Dan Olaru
I.P. Iasi

1. Introducere

Cele mai utilizate metode de realizare a comunicatiei seriale, mod de transmisie a datelor numerice cu evidente avantaje, sint :

a) folosirea a doua porturi (unul de intrare iar celalalt de iesire) si efectuarea de catre microprocesor a operatiilor de serializare/deserializare. Aceasta metoda, cea mai economica din punct de vedere hardware, prezinta dezavantajul utilizarii ineficiente a microprocesorului si este folosita cu precadere in sistemele dedicate ieftine.

b) utilizarea unui circuit VLSI dedicat (I8251, Z80-SIO, etc.)

In continuare vom propune o a treia varianta de implementare a comunicatiei seriale prin folosirea unui circuit timer (I8253, Z80-CTC). Metoda propusa asigura o utilizare superioara a mP comparativ cu varianta a), in absenta unui circuit de tip USART, si permite eliminarea circuitelor pentru generarea tactului de comunicatie. Circuitul timer poate fi folosit in acelasi timp si pentru alte aplicatii tipice ale acestuia.

2. Principiul metodei

Vom considera o linie de comunicatie semiduplex cu viteza de 1200 bauds conectata la un microsistem cu frecventa ceasului de 2 Mhz. Transmisia este de tip asincron : 1 bit START, 8 biti date, 1 bit STOP, fara control de paritate. Modalitatea de conectare a circuitului 8253 [2] este cea din figura 1.

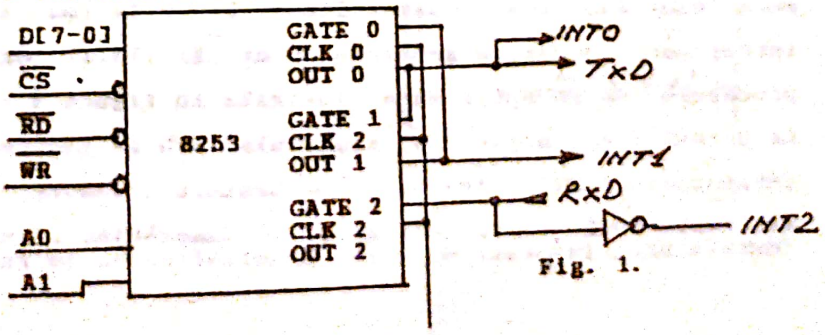


Fig. 1.

Iesirea OUT 0 (numaratorul #0) este conectata la linia de emisie TxD, iar intrarea GATE 2 (numaratorul #2) la linia de receptie RxD. Daca f este frecventa de transmisie si F este ceasul procesorului din sistem, vom nota cu n0=F/f numarul de perioade de ceas procesor cuprinse in intervalul necesar transmisiei/receptiei unui bit. Selectarea unui numarator se face prin transmiterea cuvintului de control corespunzator (fig. 2)[2].

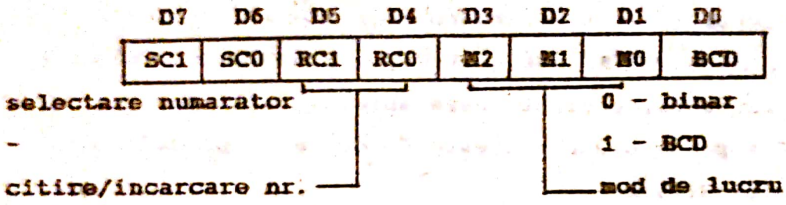


Fig. 2.

Pentru emisie se va utiliza numaratorul #0 in vederea stabilirii duratei Low a semnalului pe linie si numaratorul #1 pentru durata High a aceluiasi semnal (TxD). Se vor programa nr. #0 in modul 1(programmable one-shot) iar nr. #1 in modul 4 (software triggered strobe), inainte de inceperea propriu-zisa a emisiei, si apoi in modul 1. In vederea receptiei numaratoarele se vor programa dupa cum urmeaza : nr. #2 in mod 4, nr. #1 in mod 1, nr.#0 in mod 4. In cele ce urmeaza vom considera pe rind cele doua situatii.

2.1 Receptia

Procedura de receptie este declansata de primul front negativ al semnalului RxD. Esantionarea liniei RxD este realizata prin intreruperi succesive generate de nr. #1 (INT1). Organigrama procesului de receptie este ilustrata in figura 3.
La primul front negativ al semnalului RxD se genereaza o intrerupere (INT2). In rutina de tratare a acestei intreruperi se mascheaza INT2, se incarca nr.#2 cu Nmax>=8*n0 si nr. #1 cu n0/2 (durata unei jumatati de bit). Deserializarea se realizeaza prin

determinarea succesiva a bitilor CH[I]. Initial CH[0]=0 si variabila contor I=-1. Dupa anularea continutului nr. #1 se genereaza INT1. Daca I=-1 si continutul nr. 2 este diferit de Nmax rezulta ca nu a fost vorba de un bit de start (pe durata Tbit/2 a revenit in starea High). In aceasta situatie se demaschează intreruperile si se revine in programul principal. Daca in schimb continutul nr. #2 nu este modificat, se va reincarca nr. #1 cu n0 si se revine in programul principal. In situatia I=-1 se preleveaza continutul nr. #2 calculindu-se n2 (variatia continutului fata de situatia precedenta). In functie de valoarea n2 se determina CH[i]. Procesul continua pina se determina toti cei 8 biti de date si bitul de STOP.

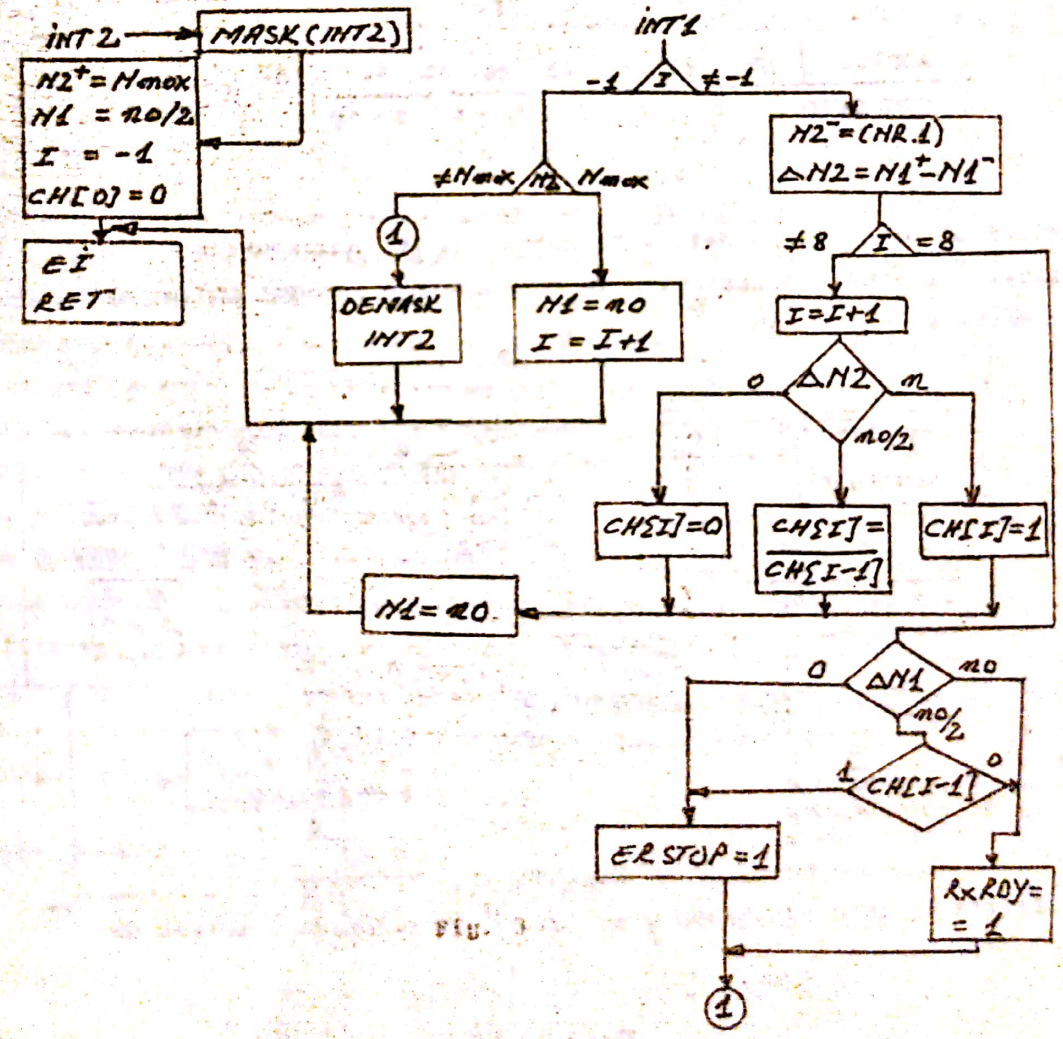


Fig. 3

2.2 Emisia

Vom considera caracterul de serializat de forma urmatoare (fig. 4) :

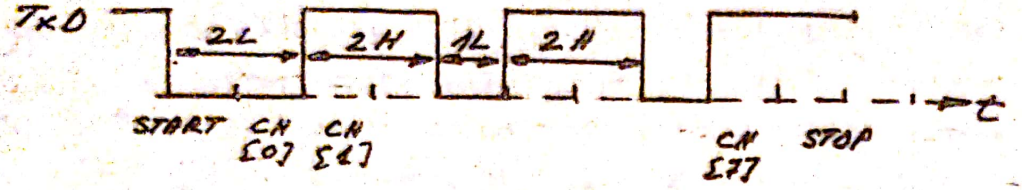


Fig. 4

In exemplul considerat pe linia TxD vom avea o succesiune de forma : L=2 - H=2 - L=1 - H=2 etc. (L=low, H=high).

Prima parte a al algoritmului (FORM) isi propune determinarea sirului de valori corespunzator caracterului de transmis.

Se va incarca un buffer (LUNG) cu valorile determinate (fig. 5a).

ADRESA	LUNG	+1	+2	+3	+4	+5	+6
CONTINUT	2	2	1	2	1	1	FF

Fig. 5a.

Sfirsitul blocului de date este marcat cu FF(corespunzator bitului de STOP). Organigrama de completare a acestui buffer este prezentata in fig. 5b.

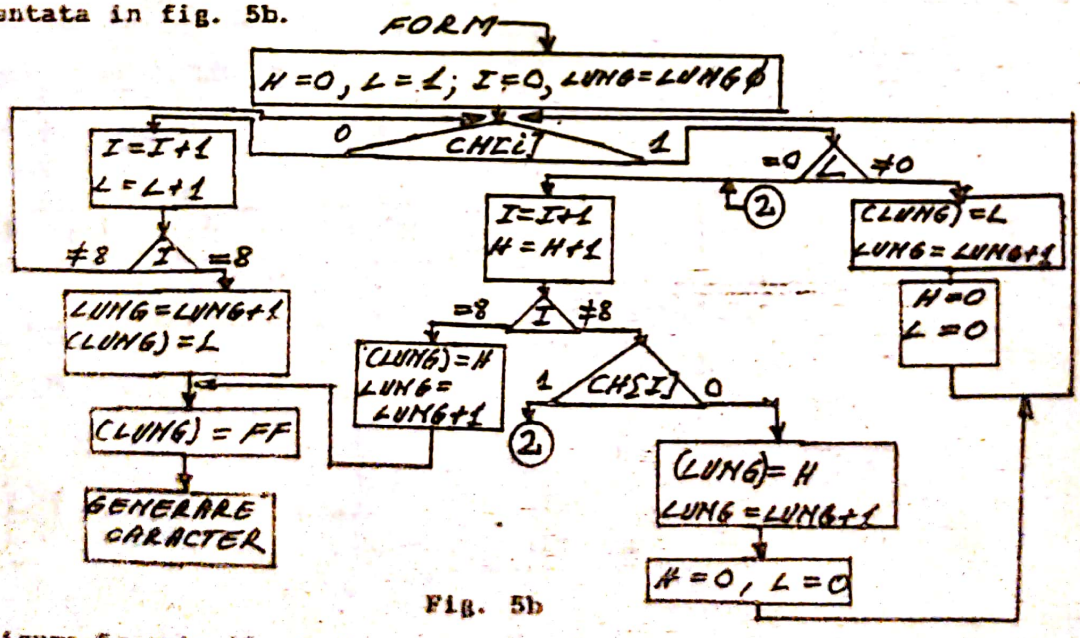


Fig. 5b

In figura 5c este ilustrata organigrama de generare a formei de

unda pentru exemplul considerat.

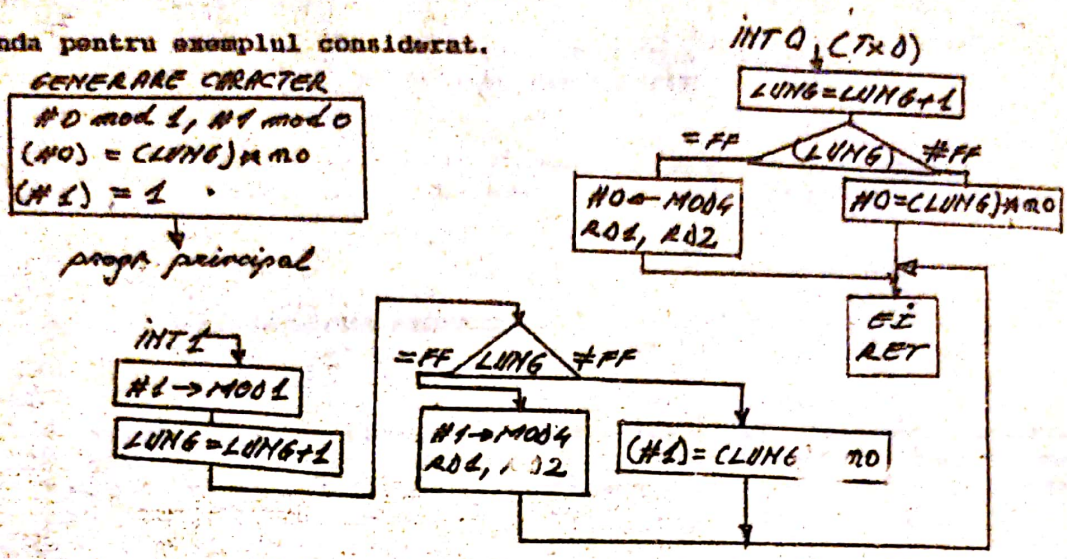


Fig. 5c

Initial se programeaza numaratoarele $0 si $1 in modul 1, respectiv 0, dupa care se incarca nr. $0 cu n0 * L (L fiind prima valoare din stiva LUNG) si nr. $1 cu 1. Dupa un timp egal cu perioada ceasului 0 se obtine un front pozitiv la iesirea INT1. Tratarea acestei intreruperi consta in incrementarea variabilei LUNG si testarea continutului locatiei (LUNG). Daca acesta este diferit de FF, se reprogrameaza nr. $1 in modul 1 si se incarca cu valoarea H * n0. La frontul pozitiv al semnalului TxD (INT1) se activeaza nr. $1 , se blocheaza nr. $0 si apoi se incarca cu noua constanta L * n0. Procesul continua pina se intilneste sfirsitul caracterului (FF in buffer), moment in care se trece numaratorul respectiv in modul 4. In acest mod se asigura starea High la iesire si se blocheaza functionarea astabila a grupului format din nr. $0 si nr. $1.

Bibliografie

[1] Th.R. Blakeslee - Proiectarea cu circuite logice MSI si LSI Ed. Tehnica - 1988
[2] *** - Catalog INTEL 1985

SIMULARE GRAFICA A UNUI PLANIFICATOR DE ACTIUNI PENTRU ROBOTI

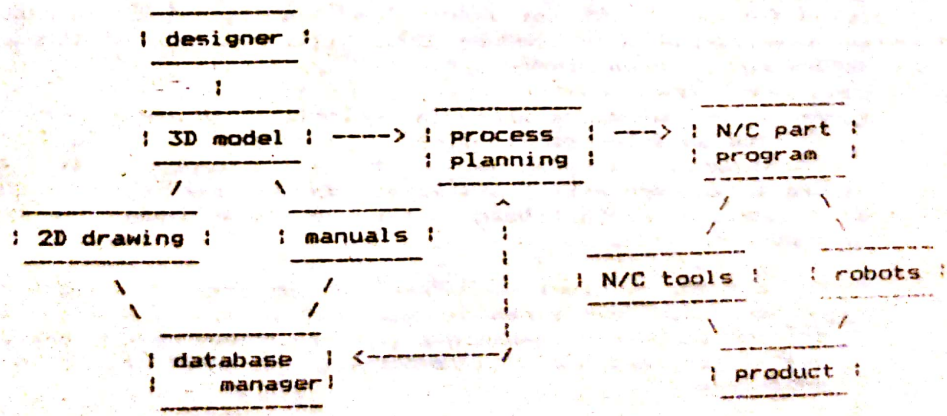
ing. S. Ticrea A.C.O.M.I.N. Cluj
asist. ing. C. Rusu I.P. Cluj

1. PROBLEMA ABORDATA

Obiectul lucrarii de fata il constituie realizarea unui produs program in domeniul roboticii axat pe generarea de planuri de actiuni si modelarea lumii exterioare pentru un robot din lumea cuburilor.

Tema planificarii robotilor a fost abordata inca din anii '70 prin proiectele universitatilor M.I.T. si Standford si care treptat a inceput sa preocupe mai toate institutele de cercetari din intreaga lume. Au fost realizate astfel asa numitele "robot planning systems" printre care se numara PLANNER, DCOMP, RSTRIPS si altele, sisteme bazate pe calculul predicatelor si pe teoria cadrelor.

Pornind de la aceste considerente si de la rezultatele obtinute in acest domeniu, am considerat utila realizarea unui planificator de roboti inteligent intr_un sistem grafic de simulare pe care l-am denumit RSRP (Romanian System of Robot Planning). Acest produs se doreste a fi o prima componenta a unui sistem CAD-CAM destinata cercetarii si testarii diferitelor tipuri de ansambluri robot capabile de a gasi singure secventa de actiuni pentru atingerea unui anume scop. Un sistem CAD-CAM are in general o structura de tipul celei descrise in schema bloc de mai jos :



Se observa clar delimitarea si interactiunea celor patru module ce alcatuiesc acest sistem: modulul CAD, planificatorul propriu-zis, baza de cunostinte a sistemului si modulul CAM. Lucrarea de fata descrie implementarea primelor trei module in domeniul lumii cuburilor abordind probleme de CAD, infografica, inteligenta artificiala si calcul al predicatelor. Totodata a fost realizat un produs soft independent destinat interfatarii sistem-utilizator in conditiile grafice oferite de calculatorul utilizat.

Sistemul creat astfel are urmatoarele facilitati:

- posibilitatea reprezentarii in trei (3) dimensiuni a mediului
- posibilitatea construirii iteractive a lumii de cuburi impreuna cu atributele sale
- facilitati de salvare/restaurare a scenei
- posibilitatea elaborarii unui plan de actiuni (descris pe baza unui limbaj formal) pentru atingerea unei anumite situatii pozitionale a cuburilor.

2. MODULUL C.A.D.

2.1. Alegerea modelului de reprezentare

Prima problema abordata in modul a fost gasirea modelului de reprezentare a lumii inconjuratoare. Modelarea mediului si manipularea acestui model reprezinta cheia modulului CAD al oricarui produs program. Pe de o parte un model trebuie sa fie astfel ales incit sa asigure o mare flexibilitate in manipularea sa. Pe de alta parte trebuie astfel ales incit sa inglobeze o cantitate suficient de mare de informatii pentru a reprezenta obiectele din jur cit mai exact. Am ales pe baza acestor considerente modelul hybrid-edges (1988) [3]. Modelul sistemului este structurat pe doua nivele astfel:

- pe primul nivel se plaseaza obiectele primare. Aceste obiecte se afla descrise intr-un sistem propriu de coordonate, codificat pe suportul fizic extern (disketta) Ele alcatuiesc astfel biblioteca de componente a sistemului avind posibilitatea astfel de a o extinde cu noi tipuri de obiecte.
- pe al doilea nivel se plaseaza instantele obiectelor primare ce reprezinta de fapt obiectele propriu-zise din jurul nostru. Instantele contin atributele ce individualizeaza instanta si o legatura la obiectul primar ce a generat-o.

Pentru actualizarea permanenta a nivelului doi al modelului au fost implementati operatori de "creare" si "stergere" ce permit crearea, respectiv stergerea, unei instante.

2.2. Reprezentarea 2D

Modelul de reprezentare ales se bazeaza pe utilizarea unui sistem de coordoate rectangular de tip RHS. Pentru obtinerea unei viziuni tridimensionale asupra scenei a fost folosita proiectia paralela dupa o directie, pe planul XOY. Sistemul fiind iteractiv, exista posibilitatea deplasarii pe toate cele trei axe de coordonate prin intermediul unui cursor grafic tip sageata. In permanenta este afisata pozitia reala din "world coordinate" (WC) in care ne situam. Proiectia unui corp a tinut cont de existenta fetelor ascunse eliminind astfel calcule in plus. Deasemenea proiectia respecta algoritmul "pictorului", obiectele fiind proiectate incepind de la coordonata z cea mai mica spre cea mai mare. Pentru o cit mai buna viziune, am asigurat posibilitatea detalierii unor parti din scena prin facilitati de zoom oricit de adinc, revenirea la viziunea de ansamblu a scenei realizindu-se urmind drumul invers detalierii, sau direct. Aceste facilitati m-au obligat la realizarea si a clippingului 3D a scenei dupa metoda Cohen Sutherhand [4].

3. BAZA DE CUNOSTINTE GLOBALA

Dupa cum este lesne de presupus dupa cele prezentate in capitolele anteriare, aceasta baza de cunostinte trebuie sa contina atit informatii analitice cit si formale despre lumea exteriora. Prin urmare aceasta baza de cunostinte a fost conceputa sub forma unei liste de obiecte individualizate prin atributele sale. Astfel un obiect are un nume, greutate, culoare si informatii formale privitoare la pozitia acestuia in raport cu celelalte obiecte. Analitic, fiecare obiect este bine determinat prin pozitionarea precisa in WC a originii acestuia din sistemul cartezian propriu. Aceste informatii fac parte din baza de cunostinte dinamica datorita operatiilor de creare/stergere. Tot din baza de cunostinte dinamica mai face parte si descrierea prin predicate formale a starii curente in timpul procesarii planificatorului.

Cealalta parte a bazei de cunostinte o reprezinta cea statica. Din ea fac parte pe de-o parte descrierile starii initiale si a celei finale (scop) sub aceeasi forma cu cea curenta, iar pe de alta parte descrierea regulilor folosite de mecanismul de planificare. In fine, baza de cunostinte statica mai contine un tabel in care fiecarui predicat formal ce caracterizeaza relatia intre obiecte i se atribuie un cod folosit in prelucrarile ulterioare de catre analizorul sintactic.

Scanned with OKEN Scanner

4. PLANIFICATORUL PROPRIU-ZIS

4.1. Descriere formalism

Multe din facilitatile oferite de sistemele ce implementeaza inteligenta artificiala implica solutii ale unor probleme bazate pe planuri de actiuni si luarea unor decizii intr-un timp scurt. In domeniul roboticii problemele tipice care se pun sint de fapt obtinerea scopurilor prin secvente de actiuni. Aceste secvente vor sta la baza programarii implicite ulterioare a robotului prin executia efectiva a miscarilor. Prin acceptiunea de "problema robot" intelegem un robot care are de indeplinit actiuni intr-un univers simplificat. Caracteristica principala a unui limbaj implicit consta in aceea ca aceste "actiuni de indeplinit" descriu actiunile pe care robotul trebuie sa le indeplineasca si nu miscarile necesare pentru realizarea acestor actiuni.

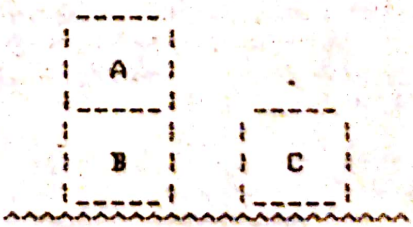
In lumea cuburilor universul robotului il reprezinta citeva cuburi denumite situate pe o masa, sau unul peste altul si dintr-un robot dotat cu o mina ce prinde si muta corpuri. Sintetizarea actiunii pentru robot poate fi solutionata printr-un sistem de productie axat pe o baza de date globala care sa oglindeasca situatia universului din care robotul face parte si regulile ce implica actiunile. O astfel de regula are in general forma:

C1 /\ C2 /\ ... /\ Cm ---> A1 /\ A2 /\ ... /\ An ,

unde Ci - conditii si Aj - descrieri de actiuni care se considera a fi adecvate de executat in situatile descrise de reguli. Datele astfel reprezentate sint apoi manipulate de catre un interpretor care compara conditiile fiecarei reguli (productii) cu descrierea situatiei curente pina cind aceasta va identifica scopul propus. Se observa astfel ca sistemul de productie ofera un control flexibil al operatiilor.

In cazul de fata descrierea starilor si a scopurilor poate fi facuta prin intermediul predicatelor. Se definesc astfel pentru lumea cuburilor predicatele:

CLEAR(x),
ON(x,y),
ONTABLE(x),
HOLDING(x),
HANDEMPTY.



Descrierea unei stari de moment se bazeaza tot pe formulele logice predicative. Spre exemplu situatia de mai sus poate fi descrisa prin conjunctia predicatelor

ON(a,b) /\ ON(b,c) /\ ONTABLE(c) /\ CLEAR(a) /\ HANDEMPTY.

Avind aceste considerente de baza am ales ca reguli in sistemele de productie regulile STRIPS-ului [1,2], care au trei componente:

DN

- lista pre: conjunctie de predicate reprezentind conditiile

- lista delete: conjunctie de predicate ce nu vor mai fi adevarate dupa aplicarea regulii

- lista add: conjunctie de predicate ce vor deveni adevarate.

Strategia de cautare abordata este una de tip backward in care conditia aplicabilitatii regulii este aceea de a exista in expresia situatiei curente un predicat ce are corespondent in unul din predicatele listei "add" a regulii. Noua stare curenta este creata prin regresia celorlalte predicate din fosta stare curenta prin instantierea regulii F, conjugindu-le cu aceasta. (Definirea formalismului regresiei este prezentat in [2]).

4.2. Mod de abordare

Descrierea starii finale (scop) cade asadar in sarcina utilizatorului care trebuie sa furnizeze lista de conjunctii de predicate ce caracterizeaza aceasta stare. Pe baza predicatelor introduse in urma unei analize sintactice si semantice este creata sub aceeasi forma lista starii curente si cea a starii initiale care se preia din situatia existenta creata interactiv de utilizator. Algoritmul planificatorului se bazeaza pe constructia unei stive de scopuri continind predicate scop ce se obtin treptat pe masura ce algoritmul backward avanseaza. Aceasta stiva scop se construieste astfel:

- se pleaca de la unul din predicatele din scopul compus ce regresat printr-o regula da "true".

- conform modalitatii de gasire a subscopului se vor adauga pe stiva in plus regula impreuna cu predicatele din lista "pre" a regulii B gasite. O astfel de adaugare la stiva reprezinta de fapt crearea unei derivatii pe stiva, derivatie care va incerca sa rezolve predicatul scop aflat la baza ei. Noile predicate scop obtinute se vor trata in acelasi mod, considerindu-se un predicat ca fiind rezolvat atunci cind toate predicatele de sub el (stiva crescind in jos) au fost la rindul lor rezolvate, sau in cazurile in care predicatele din derivatia sa apartin, toate, starii initiale. Determinarea regulii B rezolvitoare se face prin baleierea de catre procesorul de inferenta a setului de reguli implementat. O regula B este gasita rezolvitoare cind sint indeplinite anumite conditii dintre care cele mai importante ar fi:

* predicatul regreseaza "true" prin regula
* predicatele (restul) din lista "add" exista in descrierea starii curente dar nu si in cea initiala
* predicatele din lista "pre" nu exista printre cele in curs de rezolvare pe nivelele inferioare ale stivei.

Scanned with OKEN Scanner

Exista insa posibilitatea ca nici o regula sa nu rezolve predicatul analizat. Aici intervine "cheia" algoritmului: posibilitatea transferului pe pozitia predicatului nerezolvat a unui alt predicat, asa numit "ajutator", din starea finala si care devine noul predicat scop. Exista doua posibilitati:

1) in urma rezolvarii acestui predicat revenirea la predicatul initial si regresia acestuia sa ofere satisfactie

2) regresia sa dea "fals" caz in care se incearca un nou transfer. In cazul in care nu mai este nimic de transferat derivatia se sterge de pe stiva si se continua analiza predicatului de la baza derivatiei din locul de unde s-a ramas. De mentionat ca algoritmul este capabil sa restaureze contextul unei derivatii, nu prin apel recursiv, ci repetitiv.

Algoritmul se termina cind predicatele din conjunctia starii finale sint rezolvate in totalitate. In caz contrar planificarea este imposibila existind prea putine informatii referitoare la starea finala.

5. SISTEMUL DE MENIURI GRAFICE UNIVERSAL

Produsului i s-a atasat un sistem de meniuri grafice de tip pop-up, pull-down dupa modelul Borland. Acest sistem este capabil sa isi gestioneze singur resursele ecran si in plus constructia meniurilor se realizeaza pe baza unui fisier furnizat de utilizator. Acest fisier contine codificat arborele de meniuri intr-o parcurgere depth-first, coltul stinga sus al primei comenzi pop-up si latimea meniului pop-up. Sistemul este livrat sub forma de fisier .OBJ cu posibilitatea "legarii" acestuia la aplicatia dorita.

6. CONCLUZII

Lucrarea abordeaza problematica generarii planului de actiune a unui robot in lumea cuburilor cautind solutia optima de atingere a unui scop final pornind de la o stare initiala data. Pachetul de programe este bazat pe o grafica iteractiva ce permite utilizatorului descrierea starilor initiale, cele finale fiind specificate cu ajutorul predicatelor formale. Acest program a fost implementat in limbajul C pe microcalculatoare compatibile IBM-AT, obtinindu-se rezultate de genul celor prezentate in figura 1 ce prezinta descrierea unei stari initiale date si in figura 2 ce prezinta calea optima gasita.

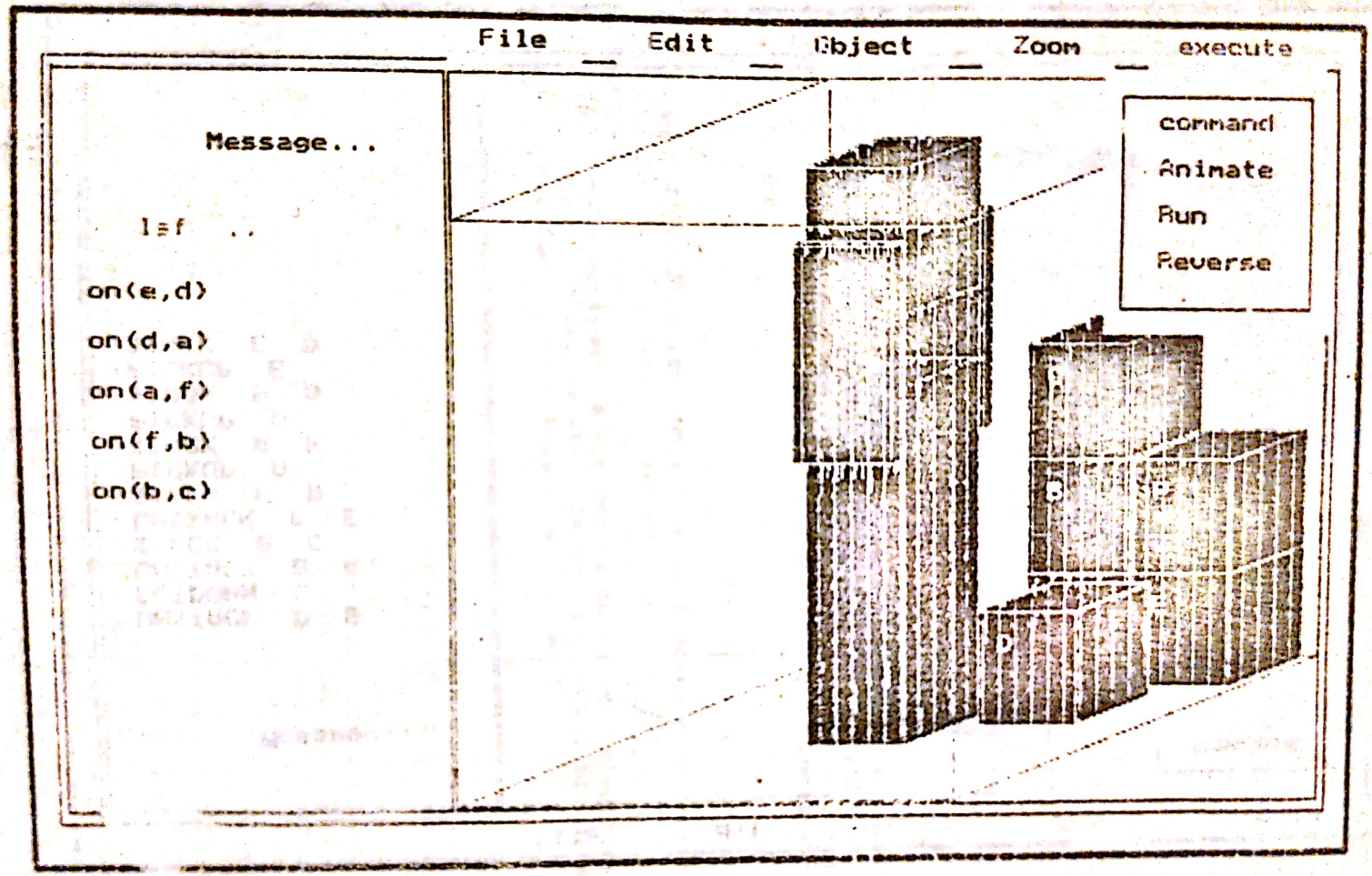


fig. 1

Scanned with OKEN Scanner

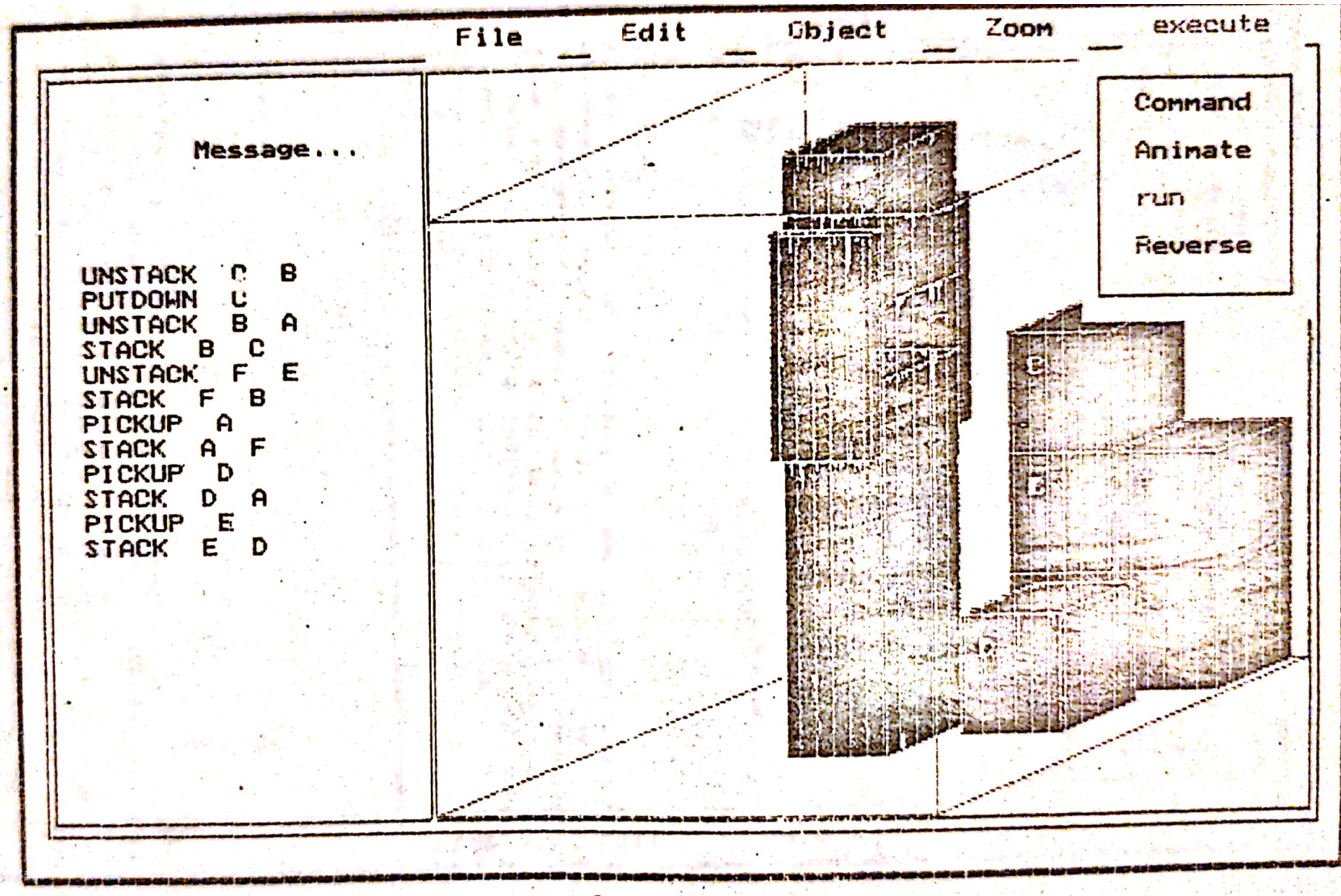


fig. 2

Scanned with OKEN Scanner

7. BIBLIOGRAFIE

[1] Niels Nilsson — Principles of Artificial Intelligence
Springer Verlag New-York 1982

[2] Patrick Winston — Artificial Intelligence
Addison-Wesley 1977

[3] * * * — Colectia revistelor CAD 1988, 1989

[4] Dorian Dogaru — Elemente de grafica 3-D
Editura Stiintifica si Enciclopedica
Bucuresti 1988

Sorin Ticrea (29 august 1967 CLUJ) a obtinut diploma de inginer in stiinta calculatoarelor la Facultatea de Calculatoare a Institutului Politehnic din Cluj in anul 1991. Domeniul sau de interes major il reprezinta simularea grafica iteractiva si inteligenta artificiala.

Calin Rusu (14 ianuarie 1958 DEJ) a obtinut diploma de inginer in specialitatea calculatoare - automatizari la Facultatea de Electrotehnica a Institutului Politehnic din Cluj, in anul 1983. In prezent este asistent la catedra de Actionari Electrice si Roboti Industriali. Preocuparile sale actuale includ Dinamica si Controlul Robotilor, Sisteme Expert de Timp Real.

Scanned with OKEN Scanner

ARHITECTURA BIPROCESOR PENTRU APLICATII DE INSTRUMENTATIE

Autor: ş.l. ing. Ştefan Poli - I.P.Iaşi

1. Introducere

Tehnica modernă a măsurărilor este astăzi de neconceput fără utilizarea sistemelor computerizate care să asigure flexibilitate, adaptabilitate la cele mai diverse cerinte, preţ de cost relativ scăzut raportat la un grad înalt de performanţă. Sistemele de instrumentaţie au apărut ca o consecinţă directă a evoluţiei tehnicii de calcul şi mai ales a domeniilor microelectronicii şi a microinformaticii. Larga răspîndire a microcalculatoarelor (mai ales a celor din familia IBM-PC) a determinat, apariţia rapidă a interfeţelor de măsură, capabile să transforme echipamentul de calcul într-un complex destinat celor mai diverse aplicaţii: osciloscoape digitale, sisteme pentru analiza şi controlul vibraţiilor, analizoare Fourier, analizoare spectrale, sisteme pentru insrumentaţie analitică în chimie, instrumentaţie medicală, etc.

Structura unui sistem de instrumentaţie (fig.1) include în principiu următoarele componente: **traductoarele** ce prelevează mărimile de măsurat, **condiţionoarele de semnal** ce realizează preprocesarea analogică a semnalelor (amplificare/atenuare, filtrare, etc.), **achizitorul de date** care transformă semnalul analogic de intrare într-o mărime numerică, **sistemul de calcul** care realizează analiza şi eventual elaborează deciziile, **ieşirile analogice** care furnizează semnale prelucrate (eventual comenzi pentru sistemele în buclă închisă) şi blocul de **postprocesare** analogică ce permite interfaţarea cu **elementele de execuţie**. Evident unele module pot lipsi în cazul unor configuraţii mai simple.

Una din problemele principale ridicate de aplicaţiile de control industrial (şi chiar de cele de măsură) este prelucrarea datelor şi elaborarea deciziilor în "timp real". Sistemele de "timp real" sînt caracterizate de posibilitatea de a realiza achiziţia datelor şi prelucrarea lor într-un interval de timp suficient de scurt în raport cu evoluţia procesului controlat (sau măsurat). Pentru orice sistem de achiziţie şi prelucrare de date sau sistem de control, indiferent de cît de lent ar fi acesta, există aplicaţii suficient de lente astfel încît sistemul să se comporte ca un sistem de "timp real". Multe aplicaţii (mai ales cele de măsură) pot fi rezolvate şi "off-line" adică achiziţia, prelucrarea şi eventual elaborarea deciziilor se pot face în etape succesive dar suficient de scurte pentru a păstra un timp de răspuns acceptabil; astfel de sisteme mai sînt cunoscute sub numele de sisteme "pseudo timp real". Creşterea de viteză cerută de multe aplicaţii se poate obţine pe mai multe căi:

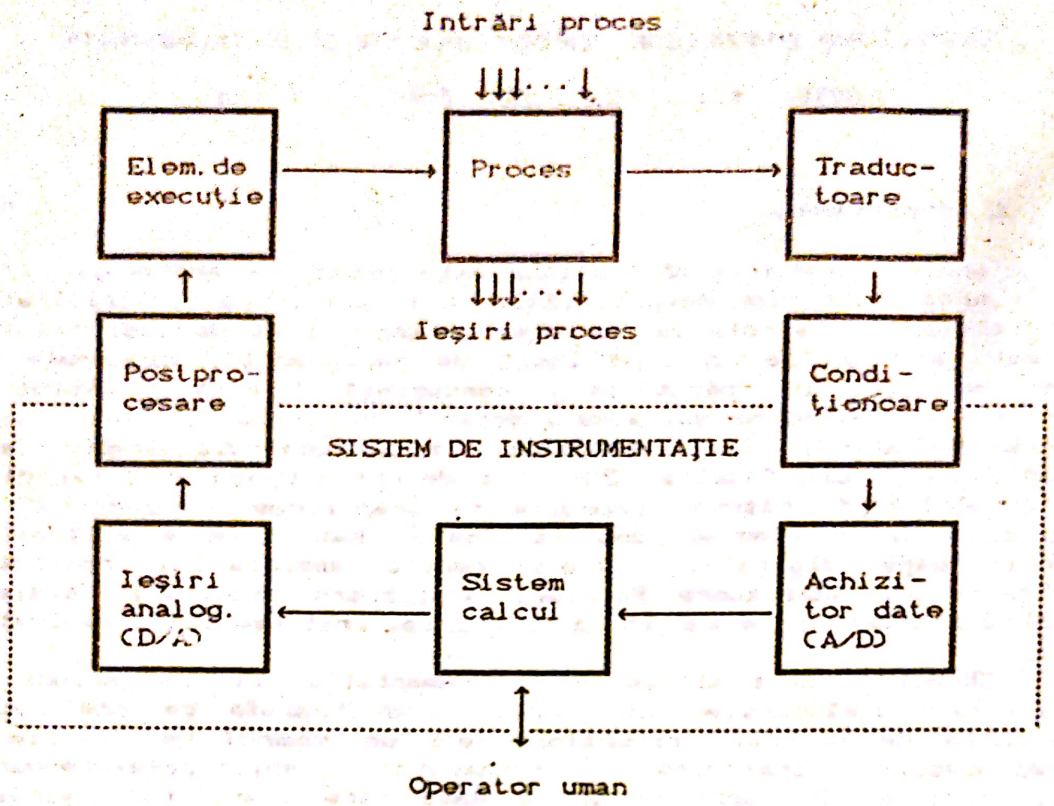


Fig.1 Schema bloc a unui sistem de instrumentaţie / control

utilizarea unor module de achiziţie suficient de rapide, eventual dotate cu unităţi de prelucrare proprii capabile să asigure şi o serie de sarcini locale de stocare şi preprocesare a mărimilor culese.

utilizarea unor sisteme de calcul performante (eventual arhitecturi multiprocesor) care să permită prelucrarea rapidă a datelor şi elaborare a deciziilor într-un timp cît mai scurt.

2. Prezentarea arhitecturii sistemului

In cele ce urmează este prezentată o arhitectură biprocesor destinată echipării sistemelor de instrumentaţie şi/sau control. Acesta permite realizarea unei staţii de lucru relativ complexe care poate îndeplini atît funcţii de interfaţare cu un proces, cît şi prelucrări proprii ale mărimilor achiziţionate. Prin conectare cu un microcalculator IBM-PC se poate obţine un complex de prelucrare distribuită extrem de flexibil.

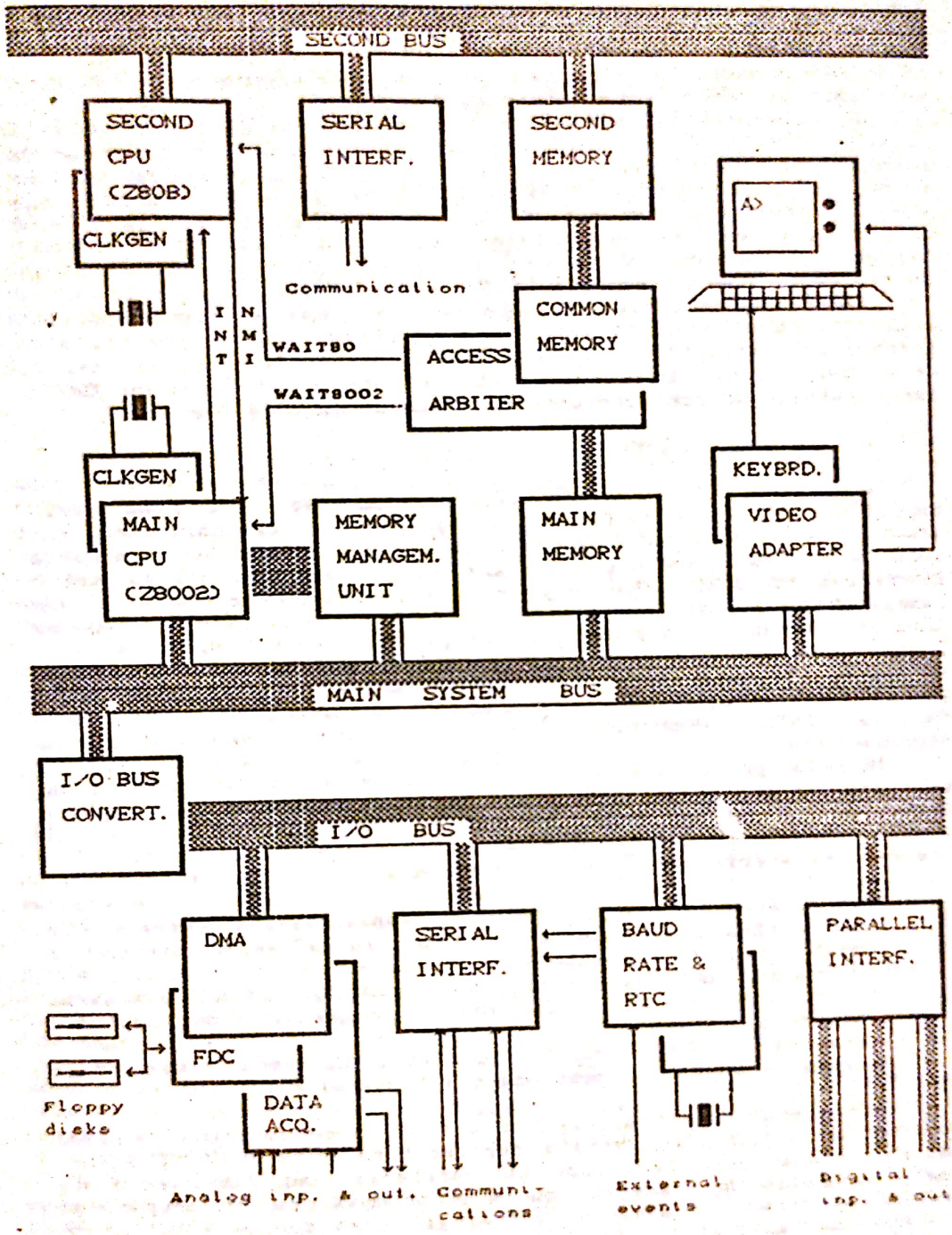


Fig.2 Sistem biprocesor pentru aplicații de instrumentație

Sistemul este compus din două structuri cu microprocesor interconectate între ele printr-o memorie biport comună (fig.2). Arbitrarea accesului se face pe baza principiului "primul venit - primul servit" cu ajutorul unui automat simplu, care pune în aşteptare procesorul solicitant dacă memoria comună a fost alocată celuilalt procesor (care o foloseşte).

Procesoarele folosite sînt microprocesorul Z8002 de 16 biţi şi microprocesorul Z80B de 8 biţi ale firmei ZILOG, dar ele pot fi substituite cu microprocesoare din familia INTEL 8086/8088 fără ca principiul interconectării să se modifice. In acest din urmă caz una din structurile folosite (sau chiar ambele) pot evolua către o arhitectură de microcalculator compatibil IBM-PC. Suportul software se bazează pe sistemul de operare CP/M (pentru Z80B) şi un sistem de operare propriu PDOS (pentru Z8002).

Funcţiile realizate de cele două sisteme au fost împărţite astfel încît să poată fi asigurat paralelismul între operaţiile de intrare/ieşire (inclusiv cele de achiziţie de date) şi operaţiile de calcul. Astfel primele au fost preluate de sistemul cu Z8002, iar celelalte au fost preluate de sistemul cu Z80B.

a. Sistemul cu Z8002

Utilizarea microprocesorului Z8002 se justifică prin facilităţile oferite de fabricant, caracteristicile sale permiţînd asemănarea cu o unitate centrală de minicalculator. O comparaţie cu alte microprocesoare evidenţiază performanţe similare cu MC6800 al firmei MOTOROLA. Uşurinţa interfaţării cu circuite atît din famila ZILOG cît şi cu circuite din familia INTEL permit alegerea unei arhitecturi de sistem aliniată la o magistrală standardizată. In cazul de faţă s-a ales o magistrală MULTIBUS simplificată (dar cu succes se poate alege şi PC-BUS/XT, sau PC-BUS/AT). Explicaţia alegerii este posibilitatea schimbării procesorului cu unul din familia INTEL 8086/8088 fără un efort considerabil de reproiectare.

Memoria principală este utilizată printr-un dispozitiv de paginare cu pagini de 4kB, care permite adresarea unui spaţiu fizic de 16 MB cu numai 64 kB de spaţiu logic disponibil procesorului principal. Acesta permite o serie de facilităţi care cresc considerabil performanţele sistemului:

- posibilitatea stocării datelor achiziţionate direct în memorie fără a fi necesară "virtualizarea" prin intermediul discului magnetic (creştere de viteză);
- dezvoltarea de aplicaţii multitasking fără evacuarea taskurilor suspendate în discul magnetic (de asemenea creştere de viteză);
- utilizarea ca RAM disc în aplicaţii de mare viteză care reclamă un mijloc de gestionare a fişierelor.

Subsistemul de intrări/ieşiri a fost conceput într-o manieră modernă utilizînd o magistrală suplimentară de 8 biţi (I/O BUS) destinată doar transferurilor cu periferia. Au fost prevăzute toate tehnicile de transfer (buclă programată, pe întreruperi şi prin acces direct la memorie) astfel încît să se poată obţine un maximum de adaptabilitate la cele mai diverse cerinţe ale periferiecelor şi aplicaţiilor. Asfel dispozitivele lente

(interfețe seriale, achizitoare lente, interfețe paralele orientate pe octet) sînt cuplate utilizînd legarea prin buclă programată și pe întreruperi, iar dispozitivele rapide (disc magnetic, achizitoare de viteză) sînt cuplate utilizînd legarea prin DMA și întrerupere la sfîrșit de transfer.

Sistemul include și controlul întreruperilor vectorizate pe 8 nivele, extensibil la 16.

Se poate observa că funcțiile destinate acestui sistem îl transformă practic într-un procesor de intrare/ieșire și achiziție de semnal pentru celălalt sistem cu care este interconectat. În afara acestor funcții el mai poate prelua și funcții de tip "coprocesor" în cazul în care celălalt procesor este mai puțin performant.

b. Sistemul cu Z80B

Destinat în principal calculului și elaborării deciziilor, acest sistem a fost redus la un minimum de resurse (practic fără resurse de intrare/ieșire). Utilizarea microprocesorului Z80B se justifică doar prin simplitatea implementării. Într-o variantă mai performantă se poate înlocui acest sistem direct cu o arhitectură de tip IBM-PC. Din considerente de viteză s-a ales un procesor dintr-o clasă superioară (6 MHz) care să asigure totuși un minimum de performanțe de calcul.

Memoria principală conține 60 kB RAM și 2 kB ROM, restul de 2 kB fiind destinați să acopere spațiul memoriei comune. Pentru ușurința depanării a fost prevazută o interfață de comunicații seriale RS232 pentru lucrul în mod local.

c. Interconectarea prin memorie biport

Cele două procesoare pot partaja o zonă de memorie comună de 2kB într-un mod perfect transparent pentru utilizator. Arbitrarea accesului se face după principiul "primul venit - primul servit"; eliminarea situațiilor conflictuale se face prin punerea în așteptare a procesorului care face o cerere și găsește resursa (memoria comună) ocupată de celălalt procespr (fig.3 și fig.4).

Memoria biport este organizată în funcție de cerințele procesorului cu magistrala mai mare (Z8000) care impune divizarea în bancuri HIGH și LOW. La nivelul celuilalt procesor accesul fiind numai pe octet acesta poate fi dirijat spre partea HIGH sau LOW doar pe baza bitului de adresă A0 (adrese pare/impare).

Sistemul cu Z8002 "vede" această zonă comună într-o pagină de memorie fizică între adresele FF0:800..FF0:FFF. Ori de cîte ori este necesar un acces în această zonă, se comută prin MMU o pagină logică nefolosită înscriind în registrul de pagină aferent valoarea OFF0H.

Sistemul cu Z80 (coprocesorul) "vede" la rîndul său zona comună între adresele F800..FFFF, după memoria EPROM (care începe la F000). Zona este în permanență accesibilă procesorului.

Pentru evitarea conflictelor "software" la comunicația între cele două procesoare a fost prevăzut un mecanism de semnalizare, implementat pe întreruperi. Astfel sistemul cu Z8002 utilizează NMI pentru cererea de întrerupere adresată de coprocesorul cu Z80, iar acesta din urmă utilizează INT (modul IM 2) pentru cererea de întrerupere adresată de primul.

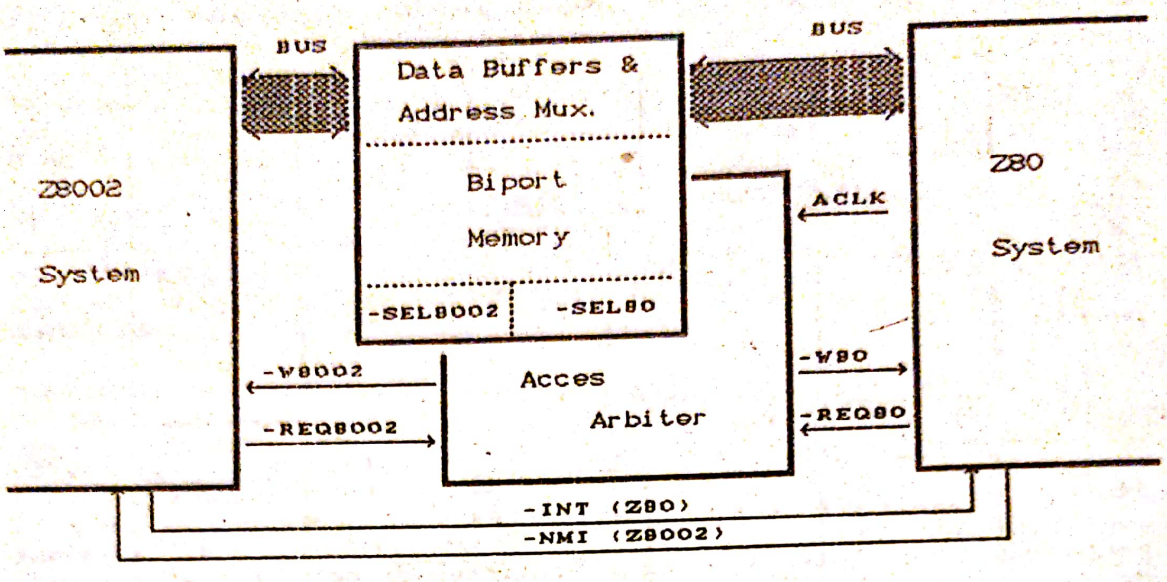


Fig.3 Interconectarea prin memorie biport

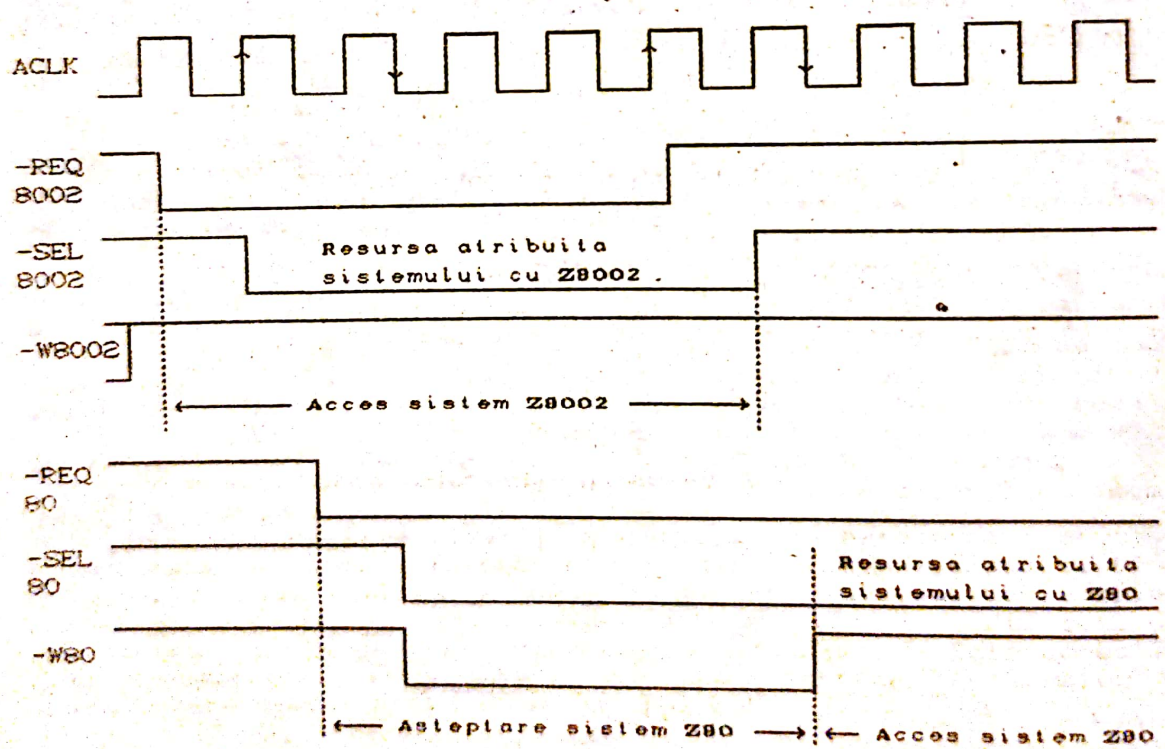


Fig.4 Arbitrarea memoriei comune

D12

Automatul de arbitrare elimină situațiile conflictuale cînd cele două procesoare cer accesul simultan la memoria comună. O soluție simplă utilizează ceasul de arbitrare (ACLK) ascuțind frontul pozitiv cererilor sosite de la Z8002 și frontul negativ cererilor sosite de la Z80B. In acest fel alegerea procesorului care va face acces în cazul cererilor simultane, depinde de primul front activ al ceasului de arbitrare (dacă frontul pozitiv soseste primul atunci se va atribui resursa procesorului 8002). O descriere posibilă a schemei de arbitrare este dată mai jos.

Secvența de arbitrare a cererilor pentru procesorul Z8002:

```
...
sel8002=w80=0;
if(req8002)
{
     sel8002=1;
     if(req80)
          w80=1;          /* Z80 trece in asteptare */
     else
          w80=0;          /* eliberează accesul Z80 */
     while(w8002);        /* așteaptă accesul Z8002 */
     access_8002();       /* are loc accesul Z8002  */
     sel8002=0;
}
...
```

Secvența de arbitrare a cererilor pentru pr cesorul Z80B

```
...
sel80=w8002=0;
if(req80)
{
     sel80=1;
     if(req8002)
          w8002=1;        /* Z8002 trece in asteptare */
     else
          w8002=0;        /* eliberează accesul Z8002 */
     while(w80);          /* așteaptă accesul Z80 */
     access_80();         /* are loc accesul Z80  */
     sel80=0;
}
...
```

3. Blocul de interfețe de instrumentație

Subsistemul de achiziție de date analogice permite o gamă largă de utilizări în aplicații pentru care sînt suficiente precizii de 12 biți și viteze de pînă la 50000 de eșantioane pe secundă. Acesta cuprinde 16 canale de intrare multiplexate și două canale de ieșire independente.

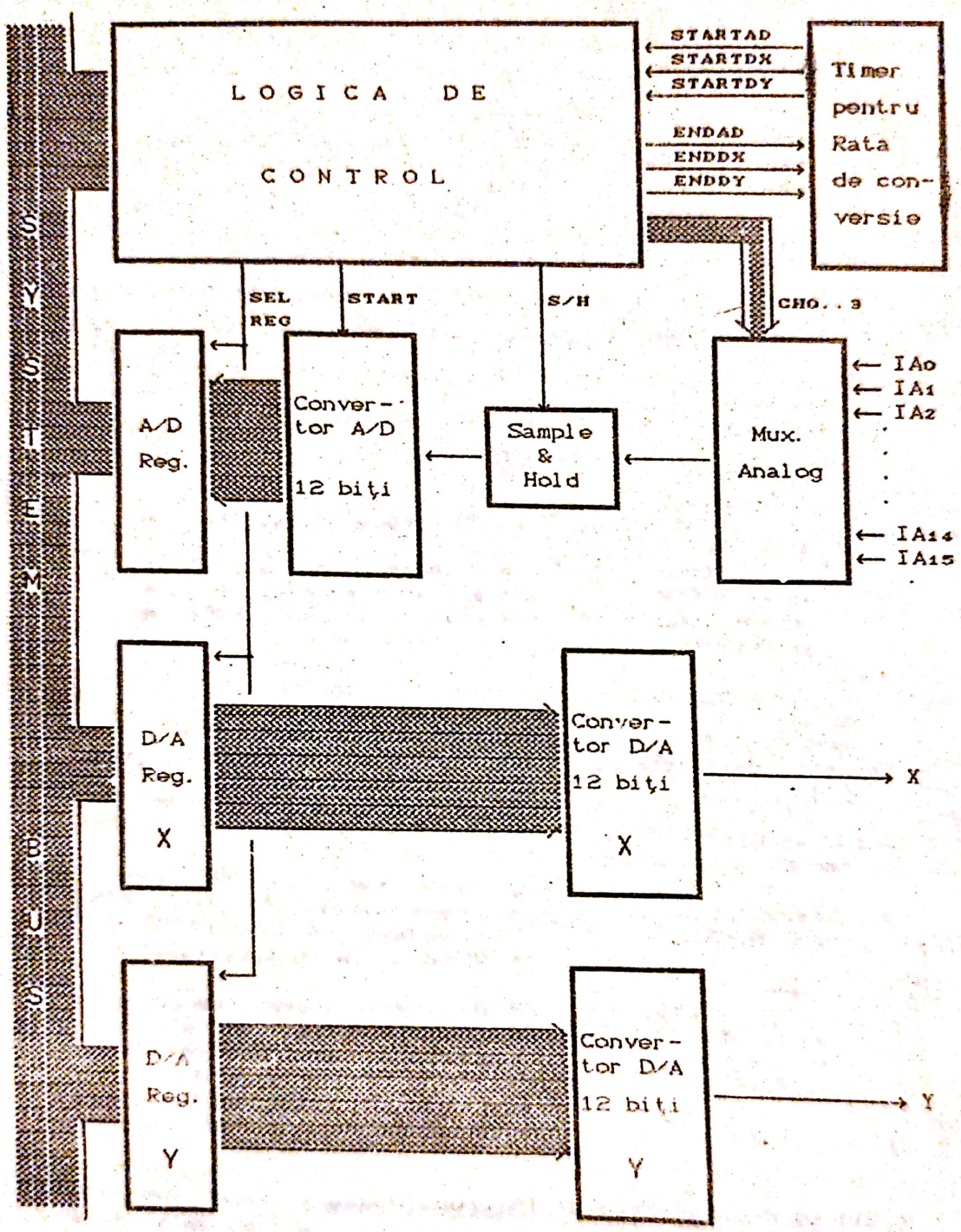


Fig.5 Subsistemul de achiziţie de date analogice

D12

Flexibilitatea subsistemului este asigurată prin alegerea unor tehnici de interfaţare adecvate. Astfel toate modurile de transfer (buclă programată, pe întreruperi şi DMA) sînt posibile ceea ce permite o bună adaptare la cerinţele aplicaţiei. Dintre caracteristicile acestui modul se amintesc:

Intrări analogice:

rezoluţie: 12 biţi ± 1/2 LSB
rată de eşantionare: pînă la 50kHz
gama de tensiuni de intrare: ± 5V
curent de polarizare la intrare: 1μA
număr de canale de intrare: 16 asimetrice multiplexate
cod: bipolar complement faţă de 2 cu ext. de semn la 16 biţi

Ieşiri analogice:

rezoluţie: 12 biţi ± 1/2 LSB
gamă de tensiuni de ieşire: ± 5V
rată de conversii: pînă la 50kHz
impedanţă de ieşire: < 100Ω
cod: bipolar complement faţă de 2 cu ext. de semn la 16 biţi

Pentru a obţine o rată de eşantionare precis controlată sistemul posedă un timer programabil pilotat de un oscilator cu cuarţ; modificarea ratei se poate face cu o rezoluţie de 1μs în gama 20μs..60ms.

4. Sisteme de achiziţie şi prelucrare de date analogice distribuite

Realizarea unor configuraţii de instrumentaţie complexe ridică probleme de interconectare deosebite. Se cunosc două soluţii cu largă răspîndire: conectarea directă (internă) pe magistrala calculatorului şi conectarea externă printr-o interfaţă de comunicaţie standardizată (RS-232, RS-422, IEEE-488).

Conectarea externă (fig.6) prezintă următoarele avantaje: poate fi configurat un sistem oricît de complex, subsistemul de achiziţie poate fi plasat la distanţă (cît mai aproape de locul de prelevare a mărimilor de măsurat), subsistemul de achiziţie poate fi interfaţat practic cu orice tip de calculator.

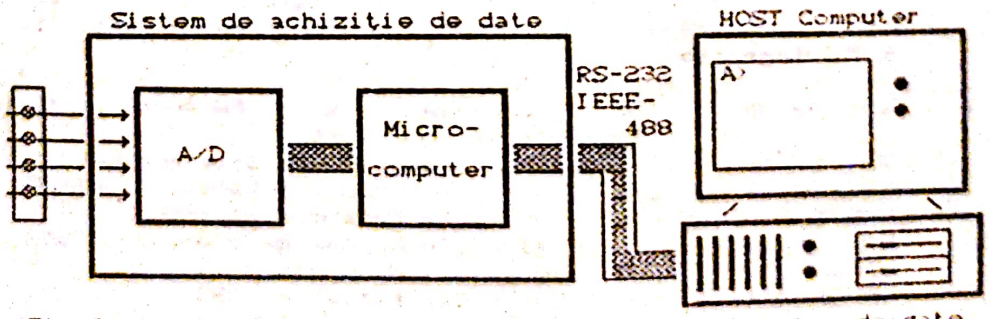


Fig.6 Conectarea externă a unui sistem de achiziţie de date

Conectarea internă (fig.7) prezintă şi ea o serie de avantaje: viteză mare datorită conectării sistemului direct pe magistrala calculatorului gazdă, preţ de cost redus, gabarit mic.

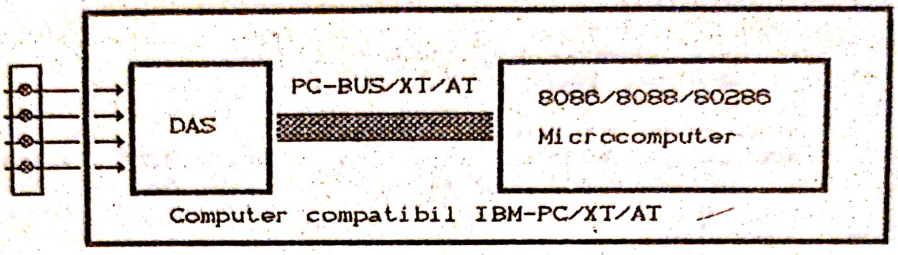


Fig.7 Conectarea internă a unui sistem de achiziţie de date

Sistemul de instrumentaţie prezentat poate fi utilizat atît de sine stătător cît şi în configuraţii distribuite. In acest din urmă caz el poate fi conectat pe unul din canalele de comunicaţie RS-232 pe care le posedă. Drept "HOST Computer" se poate folosi cu succes orice microcalculator compatibil IBM-PC.

5. Concluzii

Conceput pentru aplicaţii de instrumentaţie de uz general, sistemul prezentat a fost realizat pe parcursul a doi ani de studiu a arhitecturilor cu microprocesor şi a sistemelor de achiziţie de date analogice. Util atît în laborator cît şi în aplicaţii industriale, el poate fi reconfigurat după cerinţe.

O serie de îmbunătăţiri hardware prevăd echiparea sa cu un achizitor suplimentar pentru aplicaţii lente, dar care reclamă rezoluţii ridicate (16 biţi). Acest achizitor, din gama celor cu integrare, va permite şi utilizarea în condiţii de zgomot mai severe fără complicaţii software deosebite.

Concepţia generală permite şi utilizarea în aplicaţii de control în buclă închisă. In acest scop se încearcă în prezent dezvoltarea unui executiv de timp real care să asigure un suport multitasking eficient.

6. Bibliografie

1. BURR-BROWN — The Handbook of Personal Computer Instrumentation - 1989
2. BURR-BROWN — Integrated Circuits Data Book vol.33 - 1990
3. ANALOG DEVICES — Data Conversion Products Databook - 1988
4. ANALOG DEVICES — Linear Products Databook - 1988
5. IBM — IBM-PC/XT/AT Technical Reference - 1986
6. INTEL — Microprocessors Handbook - 1986
7. ZILOG — Z8000 Technical Manual - 1983
8. ZILOG — Microprocessors Components Databook - 1983

D12

METODE DE SPECIFICARE SI VERIFICARE A PROTOCOALELOR IN RETELELE DE CALCULATOARE

K.PUSZTAI, V.DADARLAT - Institutul Politehnic Cluj

Rezumat:

Scopul acestei lucrări este de a face o comparatie între diferitele metode folosite în descrierea si verificarea protocoalelor, folosind ca aplicatie un protocol la nivelul transport, protocol clasa 0 pentru o retea locală.

Aspecte teoretice:

O retea de calculatoare poate fi considerată ca organizată pe nivele, folosind în mod uzual clasificarea I.S.O. pentru sistemele deschise. Structura fiecărui nivel este reflectată prin specificarea protocolului propriu, adică prin descrierea relatiilor si legilor care guvernează comportarea entitătilor proprii protocolului. Protocoalele reale sînt complicate si se complică odată cu cresterea complexitătii retelei, de aceea s-a depus un mare efort pentru găsirea tehnicilor matematice si formale pentru specificarea si verificarea lor.

Printre cerintele esentiale avute în cazul descrierii unui protocol figurează: lipsa ambiguitătii, posibilitatea analizei lui la diferite nivele de detaliu si o bună structurare.

Căile de specificare a protocoalelor sînt date de limbajele formale sau, în faza incipientă de limbajul natural.

Descrierea prin limbaj natural este în general inadecvată deoarece este lungă, ambiguă iar corectitudinea si completitudinea specificării greu de verificat.

Metodele formale folosesc modele si pot fi împărtite în cîteva clase, astfel:

-modele tranzitionale, în care comportarea entitătilor protocolului este descrisă prin multimea stărilor si tranzitiilor. Aici pot fi cuprinse retelele Petri, diagramele de stare, grameticile, grafuri UCLA. Este o metodă relativ usor de învătat si apoi analizat (este si foarte folosită dealtfel), dar

pentru protocoalele complexe duce la manipularea unui volum mare de elemente ale multimii stărilor şi tranzitiilor.

-metode algoritmice, bazate pe limbajele de programare de nivel înalt (Pascal, Pascal concurent, Ada, C++), aici variabilele fiind legate de stările ce descriu comportarea protocolului

-modelele hibride, care combină avantajele celor două metode anterioare; în general ele actionează pe un model tranzitional care reflectă comportarea specifică a protocolului, tranzitiilor fiindu-le asociate module program care operează asupra variabilelor asociate stărilor. Astfel graful stărilor se simplifica deoarece mai multe stări sînt grupate într-o singură stare parametrizată.

În acest sens s-au dezvoltat limbaje de descriere specifice cum ar fi ESTELLE şi LOTOS, deja acceptate de CCITT.

În ceea ce priveste validarea protocoalelor se folosesc metode diverse, legate de metoda de descriere folosita, cum ar fi: metoda invariantilor, proceduri de reducere asupra expresiilor de transformare a modelului, demonstrarea satisfacerii de entităţi a diverselor asertiuni, generarea şi analiza tuturor stărilor sau a marcajelor într-o retea Petri etc.
Se încearcă astfel deducerea completitudinii, verificarea ciclicităţii, absenta blocării definitive sau demonstrarea altor proprietăţi specifice protocoalelor studiate.

Aspecte ale implementării

Protocolul la nivel transport folosit este deocamdată foarte simplu datorită legăturii fizice aproape perfecte existente, de aceea se poate încadra ca un protocol la nivel transport de clasa 0. Oricum pentru următoare dezvoltări ne vom baza pe existenta la nivelul retea a unui protocol X-25, impunînd considerarea unei retele orientate pe conexiune.

Rezultă că principalele servicii oferite la nivelul transport sînt mecanismul de gestiune a conexiunii şi de transfer date, fără a se trata controlul fluxului şi tratarea erorilor.

Descrierea protocolului la nivelul transport s-a făcut în următoarele forme:

-cu ajutorul unei biblioteci de functii şi proceduri Pascal care implementează primitivele de serviciu ale nivelului transport, parte din ele direct accesibile de la nivelul aplicatie-de tipul file server în acest caz.

-ca o masină de stări sub forma unei matrici avînd punctele de intrare caracterizate de predicate, actiuni şi stări.
-folosind limbajul ESTELLE (Extended State Transition

Language) - ca un modul continînd o colectie de reguli de tranzitii, fiecare regulă apartinînd unei stări şi specificînd care eveniment conditionează tranzitia, ce actiune este îndeplinită si care este starea următoare accesată. Evenimentele şi actiunile sînt module Pascal.

Protocolul elaborat contine cinci primitive de serviciu fiecare cu parametri specifici: connect, send, disconnect - de tip activ, precum şi listen, receive- de tip pasiv. Ele sînt direct legate de proceduri din biblioteca nestandard elaborată, dar folositoare aplicatiei. Pachetele elaborate sînt de tipul 'call request', 'call accepted', 'clear request', 'clear confirm' şi 'data'. O conexiune poate fi în una din următoarele şapte stări: idle wait queued es' lished send receive disconnect.

Folosind prima metoda de specificare am creat o colectie de proceduri şi functii Pascal ce implementează entitătile protocolului- parte din ele direct accesibile din programul utilizator. Altele, cum ar fi rutinele de întrerupere scrise în limbaj de asamblare sînt specifice şi descriu fenomene dinamice (packet arrival, counter clock).

Principala structură de date folosită o constituie descrierea conexiunii şi se realizează printr-o înregistrare Pascal cu cîmpuri precum: adresa locală şi adresa destinatie, starea conexiunii, ceas.

Aplicatia fiind de tipul file server se prevăd şi subprograme pentru gestiunea nivelelor inferioare, ele realizînd interfata dintre nivelul transport şi nivelul inferior.

Folosind a doua cale de descriere, am construit o matrice cu şapte coloane (stările) şi douăsprezece linii (evenimentele care schimbă starea: cinci primitive de serviciu, şase tipuri de pachete şi un ceas). Fiecare intrare în matrice are trei cîmpuri: predicatul, actiunea şi starea următoare.

Spre exemplu, dacă conexiunea este în stare inactivă dar 'ascultă' pe linie, un pachet de tip 'call request' duce la generarea unui pachet 'call accept' şi trecerea în area de stabilire a conexiunii.

Folosind module Pascal, sosirea unui pachet de date de tipul 'call request', dacă variabila 'listen' este pozitionată, se pozitionează cîmpul 'conn.state' al variabilei conexiune la valoarea 'established' şi se generează către nivelul inferior folosind rutina 'to_network_layer' un pachet de tipul 'call accept'.

Prin metoda masinii de stari finite se set arel- 'idle', stare2-'established', predicatul-'listen' unea- 'call accept'.

Folosind specificatia ESTELLE tranzitia apare astfel:

```
trans
     from idle
     to established
     when call request
     providedlisten
     priority none
          begin
           call accept
          end.
```

Concluzii

Aceste aspecte tratate fac parte dintr-un proiect de cercetare pentru dezvoltarea unui sistem expert dedicat verificării și acceptării protocoalelor de retea.

Bibliografie

Pilat F.,Cristea V. - Teleinformatica , Ed. Stiintifică si Enciclopedică Bucuresti.

Stallings W. - Data and Computer Communications , MacMillan Publish.Co

Nussbaumer P. - Téléinformatique - Ed. Presses Romandes Lausannes

Tanenbaum A. - Computer Networks , Prentice Hall Intern. Editions

Gofton P. -Mastering Serrial Communications - Sybex Publish. Comp.

Arhitectura unui sistem multiprocesor de tip cub

Sef. lucrari ing. Dan Grigoras
I.P.Iasi

1.Introducere

Exista numeroase arhitecturi de sisteme multiprocesor, a caror varietate este dictata pe de o parte de adecvanta la o anumita clasa de aplicatii, iar pe de alta de resursele tehnologice disponibile.Se cunoaste ca, cu cit un sistem de calcul este mai paralel, cu atit este mai specializat si, ca un corolar, are performante mai bune.

Totusi sistemele de calcul multiprocesor se pot imparti in doua categorii importante:

- sisteme strins conectate (tightly coupled), cind operatiile de comunicatii si sincronizare se realizeaza prin memoria comuna (shared memory);
- sisteme slab conectate (loosely coupled), cind acelasi operatii se executa prin canalele de I/E (message passing).

In primul caz operatiile critice sint rapide (la timpul de acces al memoriei), dar complexitatea schemelor creste datorita arbitrului de acces la memoria comuna. De aceea, se prefera o organizare ierarhica a spatiului de memorie comuna, tocmai pentru a reduce numarul solicitantilor de acces la un anumit nivel si deci de a simplifica schemele arbitrului. Aceasta solutie creeaza si alte avantaje: posibilitatea de a crea module care pot configura apoi sisteme dupa necesitati; structura ierarhica de tip arbore , hipercub sau clustere (fig. 1) permite o corespondenta mai buna cu numeroase clase de algoritmi paralelii se pot crea sisteme tolerante la defecte.

La un sistem slab conectat se presupune ca granularitatea paralelismului este mare, deci operatiile de calcul depasesc cu mult pe cele de comunicare sau sincronizare. Astfel, timpul total este putin afectat de interactiunile interprocesoare.

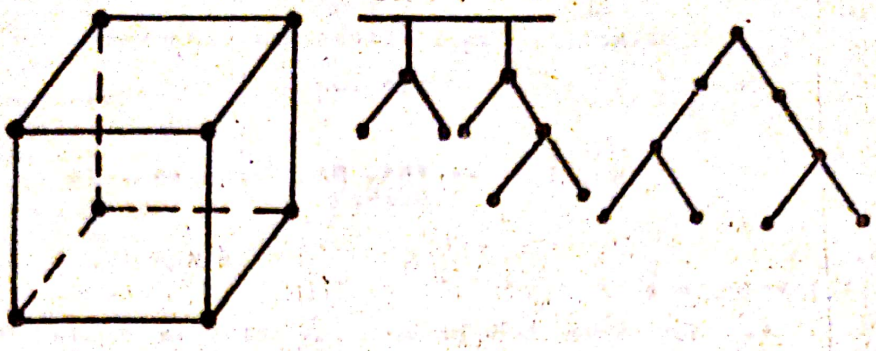


Fig.1 Topologii de sisteme multiprocesor
● - nod procesor

Mai ales in aceasta situatie, subsistemul software are un rol important in evaluarea volumului de calcul si impartirea lui echilibrata la procesoare.

Ambele tipuri de sisteme pot fi de tip stand-alone sau back-end, in acest ultim caz, apare avantajul folosirii mediului de programare a sistemului gazda. Dar si asa interfata intre host si back-end presupune un efort de programare suplimentar. In faza de cercetare se prefera aceasta solutie.

Alegerea nodurilor procesoare depinde de posibilitatile proiectantului, solutiile comerciale utilizind microprocesoarele Intel 80286, 80386,MC 68000 cu coprocesoare din aceeasi familie sau WEITEK si din ce in ce mai mult microprocesoare RISC de tip transputer T800.

2. Decizii de proiectare

Avind in vedere observatiile de mai sus am luat urmatoarele decizii:

- am optat pentru o arhitectura de tip strins conectat cu o topologie de cub (fig.2);

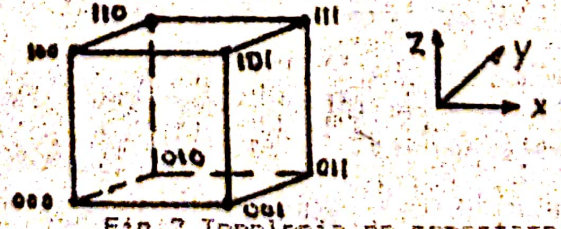


Fig.2 Topologia de conectare

O astfel de arhitectura permite executarea unei clase foarte mari de algoritmi, in speta cei ce pun in evidenta un paralelism geometric sau de tip farming. Solutiile de tip hipercub folosesc de regula comunicatia si sincronizarea prin message passing. Deci, optiunea noastra, dictata de ratiuni economice este o arhitectura hibrid.

Sistemul MARK II Cosmic Cube sau Caltech Hypercube este modelul de laborator pentru toate arhitecturile de tip hipercub. El a fost proiectat in special pentru rezolvarea problemelor de fizica, ce au un grad inalt de regularitate. Poseda 64 noduri, fiecare nod fiind un microsistem Intel 8086/8087 cu 128 KB, ce comunica cu alte 6 noduri.

Rezultatele publicate pentru executarea a 20 probleme stiintifice pe cuburi cu dimensiuni intre 32 si 128 noduri arata o eficienta medie (viteza de calcul impartita la numarul de procesoare) de 85%. Pentru anumite probleme aceasta atinge chiar 97%.

Ceea ce diferentiaza proiectul nostru de Cosmic Cube este ca acesta din urma nu poseda spatiu de memorie comuna; memoria fiecarui nod este adresata numai de procesorul respectiv. Accesul altui procesor la memoria altui nod se face prin transiterea unui mesaj via o cale de 2-4 biti cu viteza de transfer de 2-4 Mb/s.

- sistemul este considerat un back-end ce se conecteaza cu un sistem gazda . Am optat pentru un CUB_2 ca sistem gazda, datorita optiunii pentru microprocesorul Z-80. Din ratiuni de eficienta, conectarea dintre host si back-end se face prin spatiu de memorie comuna. Asa cum se vede in figura 3, calculatorul gazda este conectat direct cu 2 noduri, prin cite un segment de 1KB memorie comuna. Alegerea celor 2 noduri,in care este injectata problema de rezolvat, diametral opuse, este dictata de distanta maxima intre host si oricare alt nod.

Intr-adevar, datorita topologiei de cub, distanta maxima intre host si orice nod al retelei este 2. In plus, realizarea comunicarii intre procesoare prin memoria comuna si nu pe o cale

seriala, accelereaza acest proces.

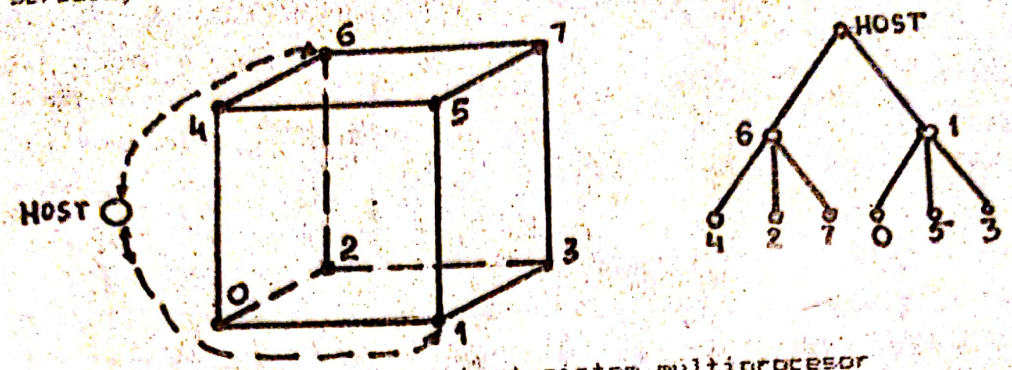


Fig.3 Conectarea host-sistem multiprocesor

3. Comunicarea intre procesoare

Pentru comunicarea intre oricare doua noduri se foloseste urmatorul algoritm: fiind date doua noduri, A si B, suma modulo 2 a numerelor lor de adresa contine 1 in dimensiunea unde numerele difera si, deci, spune dupa care dimensiune trebuie deplasata data.

Algoritmul va consta in 3 pasi (in general k):in pasul i,data va fi transmisa nodului vecin dupa dimensiunea i,daca bitul i al sumei modulo 2 este 1; daca nu, data nu se deplaseaza. Deoarece $N = 2^n$, operatia de comunicatie va consuma cel mult logN pasi.

Totusi, pot sa apara conflicte, atunci cind la un nod sosesc simultan mesaje ce trebuie transmise pe aceeasi directie.Evident, in acest caz, mesajele trebuie serializate, ceea ce duce la o scadere a performantelor. O solutie la aceasta problema consta in calcularea unei matrice generale, compusa din linii ce corespund sumelor modulo 2 dintre toate sursele si toate destinatiile. Pe baza ei, se poate stabili strategia de comunicatii, astfel incit sa se evite cit mai mult conflictele, fara a scadea prea mult perfomantele. Aceasta metoda prezinta dezavantajul unui control centralizat, care trebuie sincronizat pentru toate nodurile.

Sa consideram un exemplu : la momentul t sursa 1 are un mesaj pentru destinatia 6, iar sursa 0 pentru destinatia 5. Se produce matricea generala:

1	1	1
1	0	1
z	y	x

Se pot evita conflictele, deoarece directia y este folosita numai de sursa 1. Deci principiul metodei consta in cuplarea, in cazul dat, a transmiterii mesajelor dupa o directie fara conflict, pentru o sursa (y pentru 1) si una cu conflict pentru cealalta (x pt. 0), apoi se obtine o noua pereche de directii (x pt. 1 si z pt. 0). In acest moment mesajul transmis de nodul 1 mai efectueaza un pas.

Calculul matricei generale poate reveni procesorului host, care o va efectua in faza de partitionare si distribuire a task-urilor. Apoi, deciziile de orientare a mesajelor vor fi asociate cu momentele impuse prin procesul de calcul al fiecarui task. Astfel, chiar daca datorita naturii MIMD a controlului, este practic imposibil ca toate nodurile sa atinga simultan punctul de unde incepe o operatie de comunicatie ce ar putea genera conflicte, prin metoda propusa, acestea vor fi cu totul eliminate.

Solutia rezolvarii conflictelor la cuburile cu message-passing, este de a agrega mesajele pe directia comuna. Evident, overhead-ul impus de aceasta solutie este important.De aceea se folosesc circuite specializate de tip router.

In cazul nostru ,tot procesorul Z-80 este responsabil si pentru controlul operatiei de comunicatii.Prin urmare, in monitorul fiecarui nod se vor afla rutine de tip RECEIVE, SEND.Analiza poate continua cu stabilirea optiunii intre un protocol "store and forward", sau "cut-trough" cind fiecare byte al mesajului este transmis la urmatorul nod imediat ce ajunge. Structura sistemului de calcul proiectat impune adoptarea primului protocol.

4. Concluzii

Sistemul multiprocesor implementat consta in trei microsisteme-nod, realizate cu Z80.Considerentele economice au impus proiectarea minimala a sistemului de calcul.Oricum sistemul realizat permite testarea principalelor concepte legate de

controlul sistemelor de calcul de tip MIMD.De asemenea se pot obtine rezultate concludente privind timpii de executie ai diverselor algoritmi paraleli (numerici sau nenumerici).

Programarea sistemului se realizeaza in limbaj de asamblare, singurele facilitati oferite programatorului fiind cele ale monitorului nodului procesor.

Sistemul realizat face obiectul contractului de cercetare 1049/1989 cu SINTA S.A. si este in faza de testare/dezvoltare.

5.Bibliografie

[1] Almasi, G., Gottlieb, A.: Highly parallel computing, The Benjamin/Cummings Publishing Co, Redwood City 1989.

[2] Hockney, R., Jesshope, C.: Parallel computers, Adam Hilger Ltd, Bristol 1988.

[3] Hwang, K., Briggs, F.: Computer architecture and parallel processing, McGraw Hill, New York 1984.

Scanned with OKEN Scanner

Solutii de gestiune a resurselor in implementarea structurilor de reglare numerica pe arhitecturi de 8 biti

Octavian Păstrăvanu, Silviu Ifrim
Facultatea de Automatică şi Calculatoare
Institutul Politehnic Iaşi

1. Introducere

Cu toate că arhitecturile pe 8 biţi sînt în prezent depăşite ca performanţe de calcul prin dezvoltarea unităţilor centrale de 16 sau 32 biţi, larga lor răspîndire şi preţurile scăzute le fac încă avantajoase în realizarea structurilor de reglare destinate proceselor lente. Un inconvenient major îl prezintă însă inexistenţa unui executiv de timp real care să faciliteze proiectarea aplicaţiilor ceea ce a determinat ca nici chiar echipamente specializate (ECAROM, SPOT) să nu fie utilizate efectiv în reglarea proceselor automatizate. Pe aceste echipamente au fost implementate numai executive de timp real dedicate anumitor aplicaţii [1], [2]. Singurul nucleu de timp real destinat dezvoltării de aplicaţii generale, compatibil principial cu aceste arhitecturi este RMX [3], care prezintă dezavantaje majore legate atît de operarea sub SFDX, cît şi de lungimea codului executabil ce trebuie înscris în EPROM-uri.

În lucrare sînt analizate posibilităţile de gestionare a resurselor puse la dispoziţia utilizatorului de configuraţiile pe 8 biţi în implementarea regulatoarelor numerice pentru procese multivariabile lente. Sînt comentate avantajele şi dezavantajele fiecărei soluţii conturîndu-se totodată caracteristicile unei variante optime. S-a avut în vedere necesitatea desfăşurării concurente a următoarelor sarcini:

(S1) - achiziţia mărimilor măsurate din proces şi elaborarea comenzilor pe baza algoritmului de reglare

(S2) - stabilirea (modificarea) referinţei în conformitate cu valorile citite de la un dispozitiv de intrare

(S3) - scrierea (afişarea) valorilor variabilelor considerate de interes în urmărirea procesului.

(S1) este cea mai prioritară şi trebuie executată la intervale de timp predeterminate, egale cu perioada de eşantionare, întrerupînd (S2) sau (S3). (S3) are prioritate cea mai scăzută, pentru un dispozitiv I/O comun (de exemplu consola) (S2) întrerupe (S3).

2. Utilizarea aritmeticii în virgula mobila în implementarea algoritmilor de reglare

Fie $G_R(z) = [g_{ij}(z)]$ $i=1,p$, $j=1,m$ matricea de transfer a regulatorului discret multivariabil activat de canalele de eroare, ordinul funcţiilor $g_{ij}(z)$ depinzînd de complexitatea algoritmului utilizat. Elaborarea propriu-zisă a comenzilor se realizează pe baza ecuaţiilor cu diferenţe corespunzînd funcţiilor $g_{ij}(z)$ iar precizia cerută în efectuarea calculelor impune utilizarea aritmeticii în virgulă flotantă. Algoritmul operînd cu unităţi inginereşti, trebuie realizate conversiile unităţi CAN – unităţi inginereşti – unităţi CNA.

Cel mai avantajos mod de programare al aplicaţiei este apelarea rutinelor adecvate dintr-o bibliotecă de firmă compatibilă CP/M [4] care oferă totodată garanţii în exploatare incomparabile cu colecţiile de rutine concepute artizanal. Apelul se poate realiza din limbaj de asamblare sau direct din limbajul de nivel înalt ceea ce constituie o modalitate foarte comodă de utilizare.

Pentru cele trei biblioteci standard caracteristicile sînt:

a) FORLIB.REL – apelabilă din FORTRAN cu o reprezentare în virgulă flotantă pe patru octeţi şi durata maximă de execuţie a unei înmulţiri în virgulă flotantă de circa 3 ms.

b) FPREAL.ERL – apelabilă din PASCAL MT+, caracteristici asemănătoare cu cea anterioară.

c) MATH.REL – apelabilă din C AZTEC; toate operaţiile se efectuează în dublă precizie baza reprezentării fiind 256. Întrucît rutinile bibliotecii însăşi sînt scrise în C, durata unei înmulţiri este de circa 25 ms.

3. Realizarea driverelor pentru circuitele de interfata

Elaborarea comenzii de către microcalculatorul regulator necesită utilizarea unui ceas de timp real care să marcheze momentele de eşantionare. Programarea acestuia se realizează curent în limbaj de asamblare dar se poate face comod şi în C.

Driverele pentru lectura convertoarelor A/N şi scrierea convertoarelor N/A se implementează convenabil fie în limbaj de asamblare fie în C.

Observaţii similare sînt valabile şi pentru programarea circuitului de arbitrare a întreruperilor (dacă există în configuraţie) ,în vederea rezolvării solicitărilor simultane la resurse [5].

4. Gestiunea datelor in multitasking

Necesitatea rezolvării concurente a sarcinilor (S1), (S2), (S3) impune o disciplină în manipularea datelor specifică rutinelor reentrante. Inexistenţa

unor astfel de rutine de firmă compatibile CP/M care să execute calculele aritmetice şi operaţiile de I/O reprezintă în fond inconvenientul fundamental în exploatarea în aplicaţii de conducere numerică discretă a arhitecturilor pe 8 biţi.

Soluţia propusă în lucrare şi experimentată în numeroase aplicaţii se bazează pe următorul mod de gestiune a datelor:taskul A (provenind din modulele scrise atît în limbaj de asamblare cît şi în limbaj evoluat) este linkeditat separat de taskul B şi taskul C (provenind din module scrise atît în limbaj de asamblare cît şi în limbaj evoluat) care fac obiectul unei alte linkeditări. Cele două programe executabile comunică între ele prin variabile din blocuri COMMON (pentru limbaj FORTRAN) sau prin variabile externe (pentru PASCAL sau C). Fiecare zonă de comunicaţie este uşor de realizat în cazul blocurilor COMMON întrucît alocarea în FORTRAN este statică. În cazul alocării dinamice s-a dovedit necesară o gestiune proprie a stivei deoarece programele executabile rezultate din cele două linkeditări distincte alocă stiva în acelaşi punct.Această gestiune este un consumator suplimentar de timp la lucrul în întreruperi deoarece se forţează puncte de intrare diferite printr-o rutină anume concepută.Cele două fişiere conţinînd codul executabil sînt salvate împreună alcătuind în exploatare un singur program,eliminînd astfel inconvenientul nereentrantei rutinelor de bibliotecă.

Transferul datelor între cele două executabile este organizat numai pe variabile întregi;întrucît procesarea prin instrucţiuni UC pe dublu cuvînt este neîntreruptibilă,nu se permite alterarea variabilei datorită apariţiei unei întreruperi în timpul transferului. Se evită astfel procedeul uzual de protejare a secvenţelor neîntreruptibile prin dezactivarea şi reactivarea sistemului de întreruperi, metodă care folosită excesiv poate provoca o eşantionare neuniformă în cazul perioadelor de eşantionare mici.

5. Realizarea executabilelor promabile

În aplicaţiile industriale curente nu are sens utilizarea maşinilor CP/M, ci se realizează structuri dedicate avînd software-ul integral înregistrat în EPROM-uri.

Dacă este folosit drept limbaj evoluat C-ul în implementarea algoritmului de reglare, linkeditorul LN poate realiza direct forma promabilă prin poziţionarea corespunzătoare a secţiunilor de cod şi de date; biblioteca standard RCM.LIB asigură funcţiile sistem, iar în rutinele I/O sînt necesare unele intervenţii în funcţie de tipul dispozitivului.

Dacă se foloseşte drept limbaj evoluat Pascal MT Plus, în apelul linkeditorului LINKMT se utilizează opţiuni pentru pentru a preciza adresa codului, respectiv a datelor şi trebuiesc operate modificări în rutinele I/O.

Scanned with OKEN Scanner

înlocuind adecvat apelurile în rezidentul CP/M-ului.

Pentru FORTRAN este valabilă aceeaşi metodologie de lucru folosind linkeditorul L80.

Este deosebit de important că executabilele rezultate din module scrise în FORTRAN sau Pascal MT Plus sînt semnificativ mai reduse decît cele rezultate din module C.

6. Variante de implementare a unui regulator numeric pentru procese cu două intrări şi două ieşiri

Au fost elaborate şi testate în vederea unui studiu comparativ două versiuni scrise în limbaj de asamblare şi C, respectiv în limbaj de asamblare şi FORTRAN, ambele implementînd un acelaşi algoritm de reglare pe un echipament tip ECAROM. Experimentele au fost realizate pentru procese bivariabile modelate pe calculator analogic (de ex. [6]) sau pe instalaţii cu două mărimi de intrare şi două de ieşire (de ex. [7]). Funcţiile de transfer $g_{ij}(z)$,i=1,2 , j=1,2 ale regulatorului implementat în [6] sînt de ordin 2, iar pentru regulatorul utilizat în [7] sînt de ordin 3.

În fig.1 se prezintă organizarea generală a aplicaţiei. Comenzile se calculează în cadrul taskului TK1 lansat în execuţie la intervale de timp egale cu perioada de eşantionare T de către întreruperile ceasului de timp real: TK1 include de asemenea şi un algoritm anti-windup. TK2 şi TK3 permit interfaţarea cu operatorul: TK2 preia de la consolă valorile referinţelor pentru cele două canale, iar TK3 afişează mărimile măsurate, erorile şi comenzile. TK1 este prioritar în raport cu TK2 şi TK3 cu care comunică (în maniera prezentată în secţiunea 5) prin variabile întregi avînd funcţionalitatea unor căsuţe poştale CP. Acest mod de organizare asigură sincronizarea, excluderea mutuală şi comunicaţia între taskuri,avînd loc desfăşurarea paralelă fie a taskurilor TK1 şiTK2, fie a taskurilor TK1 şi TK3.

6.1 Varianta realizata in limbaj de asamblare si C

Pentru funcţiile de transfer $g_{ij}(z)$ i=1,2, j=1,2 de ordin 2 dimensiunea finală a programului este de 30 Ko iar perioada de eşantionare minimă cu care s-a operat a fost de 0.8s.Această variantă prezintă avantaje în programare putîndu-se intercala cu uşurinţă module scrise în C cu module scrise în limbaj de asamblare.

6.2 Varianta realizata in limbaj de asamblare si FORTRAN

Pentru funcţiile de transfer $g_{ij}(z)$ i=1,2, j=1,2 de ordin 2 dimensiunea finală a programului este de 13Ko. S-a putut opera cu o perioadă de eşantionare minimă de 0.1s.

În ce priveşte programarea, apelurile din limbaj de asamblare în FORTRAN

și invers presupun un mod mai laborios de transfer al parametrilor prin blocuri comune. Deasemenea realizarea versiunilor promabile necesită intervenții în rutinele TTYIN și TTYOT (din biblioteca FORLIB care deservesc operațiile I/O) în sensul înlocuirii apelurilor în RDOS prin secvențe de tratare proprii.

7. Concluzii

Studiul comparativ elaborat pe parcursul secțiunilor precedente a permis să se pună la punct o metodologie de proiectare și implementare a aplicațiilor de conducere numerică directă a proceselor multivariabile lente folosind arhitecturi de 8 biți. Această metodologie se bazează pe facilitățile oferit de mașina CP/M în elaborarea și testarea programului înaintea realizării versiunilor promabile.

Se poate afirma pe seama tuturor observațiilor anterioare că în ansamblu modul de programare cel mai avantajos al acestor aplicații se realizează în limbaj de asamblare și FORTRAN care conduce la viteza de lucru și imagine executabilelor cele mai favorabile. Cu toate facilitățile oferite de programarea în C sau PASCAL pentru mașinile pe 16 sau 32 biți, pentru configurațiile pe 8 biți implementările cele mai eficiente se bazează pe modulele de calcul aritmetic ale bibliotecii FORLIB.

8. Bibliografie

1. Dumitrache.I. ș.a. "Direct Digital Control of Biosynthesis Processes". Prepr. 7th Int. Conf. on Control Systems and Comp. Science, vol 1,pp 76-80,București 1987

2. Munteanu,Fl.,Muscă,Gh. "Process Control Application Based on a Dedicated Real Time Executive". Prepr. 7th Int. Conf. on Control Systems and Comp. Science, vol 1, pp 186-193,București 1987

3. *** RMX-80 User's Guide ,Intel

4.Mararu,Fl.,Atodiroaiei,M. "Programarea microcalculatoarelor în sistemul de operare CP/M",Ed. Știință,ifică și enciclopedică 1989

5.Petrescu,A. ș.a. "Microcalculatoarele Felix M18,M188,M18", Ed. Tehnică București 1984

6. Păstrăvanu,O.,Onea,Al.,Schönberger,Fr. "On-line Tuning of Digital Contrllers for Multivariable Plants with Unknown Parameters", Proc.ICATE '91, Craiova 1991 (în curs de apariție)

7. Lazăr,C.,Schönberger,Fr.,Onea,Al.,Păstrăvanu,O.,Trelea,C. "Conducerea numerică directă a unui cuptor de tratament termic", al III-lea Simpozion Național -Structuri,algoritmi și echipamente de conducere a proceselor industriale, Iași 1991

Scanned with OKEN Scanner

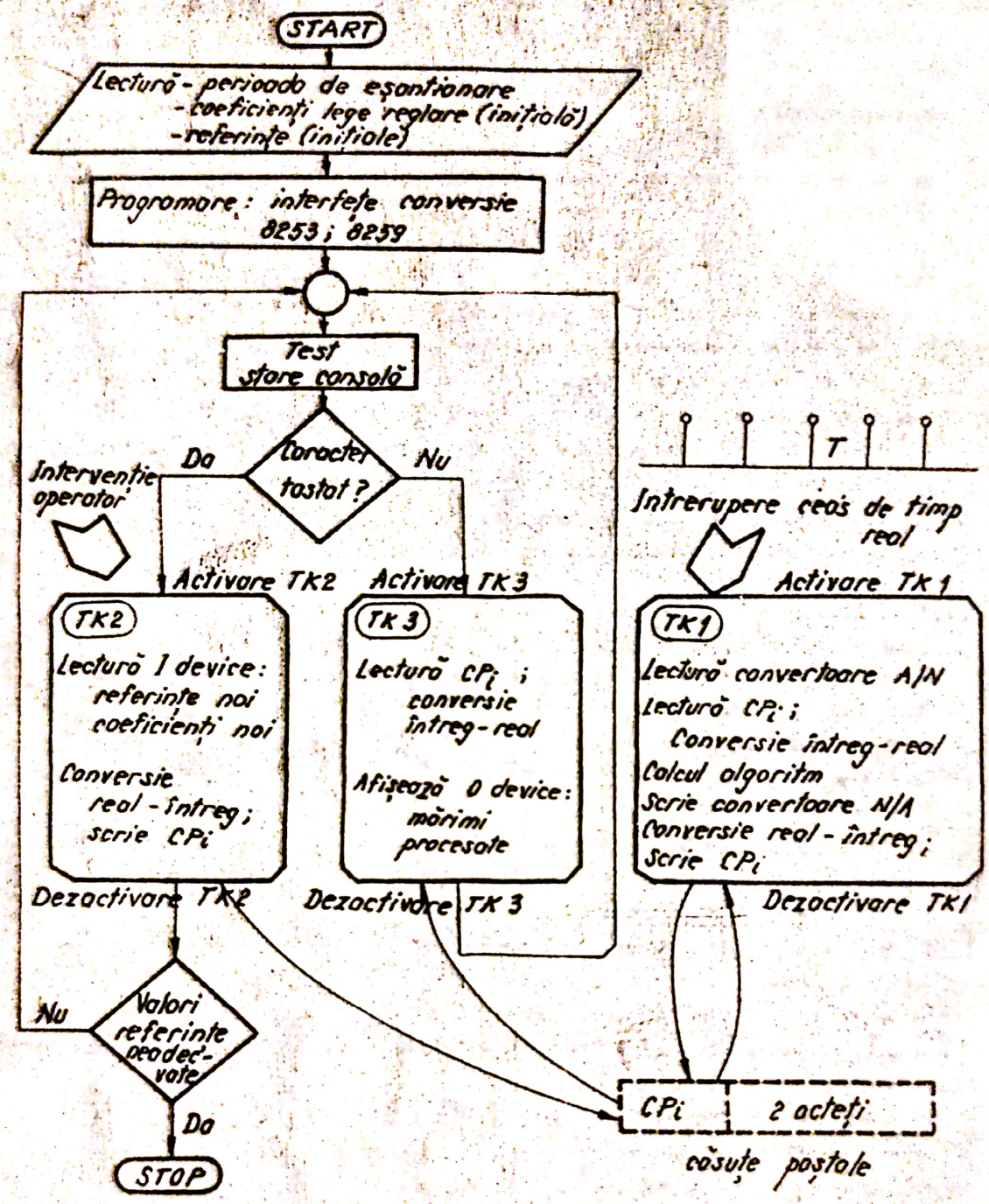


Fig. 1. Organizarea generalã a aplicaţiei de timp real

DIS

PLANIFICAREA TASK-URILOR IN SISTEMELE DE CONDUCERE A PROCESELOR

autori: as.ing.M.Litoiu,I. P. Bucuresti
Fac. Automatica

ing. M.Litoiu,I.N.M.T.Bucuresti

REZUMAT

Se considera un sistem in timp real cu un singur procesor, in care planificarea task-urilor se face dupa prioritati stabilite in faza de proiectare. Sint prezentate citeva rezultate ale autorilor, rezultate care confera instrumente simple si puternice pentru alocarea prioritatilor astfel incit, in urma aplicarii acestora, programatorul sa aiba siguranta obtinerii unui sistem inalt performant.

1.CONSIDERATII GENERALE

Sistemele pentru conducerea proceselor desfasoara o activitate ce se repeta periodic pe un interval [t,t+Mi,unde M $\neq \infty$

```
REPEAT
     PROGRAM_DE_CONDUCERE;
UNTIL <conditie>.
```

Deoarece PROGRAM_DE_CONDUCERE e alcatuit din task-uri, proprietatea de periodicitate a acestuia se transmite asupra partilor componente.Deci, intr-un sistem de conducere (ca si in alte sisteme in timp real) task-urile sint periodice:

```
REPEAT
     task 1,task 2,...,task n;
UNTIL <conditie>.
```

unde task i,i=1,n sint task-uri independente sau care se sincronizeaza in anumite puncte sau momente de timp.Dar,sistemul fiind in timp real,se impune si alta cerinta:task-urile trebuie terminate,la fiecare reluare,intr-un interval de timp bine determinat(deadline). Acest interval nu poate depasi perioada de reluare. Implementarea task-urilor se face prin intermediul unor sisteme de operare in timp real. Partea din sistemul de operare care efectueaza lansarea in executie a task-urilor se numeste planificator (sheduler). In foarte multe sisteme de operare in timp real planificarea task-urilor se face in functie de prioritatile atribuite task-urilor in faza de implementare a acestora. In capitolul urmator vor fi prezentate un model matematic si o teorema de optim pentru acest caz.

2. MODEL MATEMATIC

Se poate defini modelul ce va fi analizat ca un sistem:

(1) $\mathcal{S}$ =({Ti},{Ci},{Ii},{di},{toi}), unde

Ti,i=1,n este task-ul cu numarul i constind dintr-o secventa de program ce se cere executata periodic,a.i.:

a) Executia cu numarul k se cere la momentul

(2) toi+(k-1)*Ii; k=1,2,...;Ii este perioada lui Ti;

toi desemneaza momentul primei cereri;

b) Executia task-ului Ti cere τ_i unitati de timp.

c) Executia cu numarul k trebuie terminata nu mai tirziu de toi+(k-1)*Ii+di, $0 < \tau_i <= di <= Ii$

d) Executia task-urilor se face pe un singur procesor.

e) Task-urile sint independente, adica nu exista nici o relatie de ordine intre ele.

f) Task-urile au asignate prioritati,astfel incit,daca sint mai multe cereri de executie simultan,primul se executa task-ul cu prioritatea cea mai mare. Executia acestuia se face cu suspendarea task-ului curent(preemtibilitate), sau dupa terminarea task-ului curent(nepreemptabilitate).

Modelul prezentat corespunde unui numar important de situatii din practica sistemelor in timp real. In particular,in cazul sistemelor de conducere a proceselor,cerinta e) poate aparea ca restrictiva,deoarece in acest caz exista si relatii de ordine intre task-uri(intii se face achizitia, apoi prelucrarea , apoi se elaboreaza comanda,etc.).Aceasta situatie poate fi eliminata grupind task-urile dependente intr-unul singur . Modelul prezentat acopera si alt caz intilnit in practica programarii sistemelor si anume situatia in care programatorul foloseste doar intreruperile sistemului pentru a acoperi necesitatile de timp real. In aceasta situatie intreruperile joaca rolul task-urilor din modelul prezentat.Indiferent daca se lucreaza cu task-uri sau doar cu intreruperi,perioadele Ii ca si duratele i pot fi determinate pentru fiecare task.(Pentru task-urile care nu au o perioada naturala sau durata de executie constanta se considera cazul cel mai dezavantajos: Ii minim si τ_i maxim).

Definitia 1.Un algoritm A accepta sistemul daca planifica executia task-urilor astfel incit sa fie respectate toate intervalele di, oricare ar fi momentele initiale toi,i=1,n.

Cerinta ca sistemul sa fie acceptat oricare ar fi conditiile initiale nu este exagerata si corespunde situatiei reale din practica. Modelul (1) ales pentru analiza descrie sistemul real in situatiile cele mai dezavantajoase (cind Ii sint minime si toi maxime), momentele cind se realizeaza toate aceste situatii neputind fi cunoscute aprioric.

Definitia 2.Fiind data o alocare oarecare a prioritatilor in care task-urile Tm si Tp au alocate prioritatile k+1 si respectiv k si Ip>Im, se numeste permutare de tip P schimbarea prioritatilor task-urilor Tm si Tp astfel incit Tm sa aiba prioritatea k si Tp prioritatea k+1.

In continuare consideram ,pentru simplificare di=Ii (un task trebuie executat pina la urmatoarea cerere) si cazul preemptibil.

(3) LEMA .Daca o alocare oarecare a prioritatilor accepta sistemul ,atunci alocarea obtinuta printr-o per-

mutare P accepta, deasemenea ,sistemul $\mathcal{S}$,

Demonstratie.

Atunci cind planificarea task-urilor se face dupa prioritati si task-urile sint preemtibile ,executia task-urilor de la niveluri mai mari de nivelul k+1 nu depinde de prezenta task-urilor cu prioritati mai mici decit k+1 (prioritatea k+1 este mai mare decit prioritatea k). Task-urile de la nivelul k+1 in jos se executa in intervalele ramase 'neocupate' de task-urile cu prioritati mai mari. Deasemenea pozitia pe axa timpului a intervalelor 'neocupate 'ramine aceeasi, chiar daca se realizeaza permutari (de tip P) intre task-urile aflate mai sus de nivelul de prioritate k+1 (pastrind aceleasi conditii initiale). Task-ul ce trece de la o prioritate mai mica la una mai mare (printr-o permutare P) vafi 'acceptat ', deoarece are la dispozitie ' intervale neocupate'mai mari sau egale decit cele avute inainte de permutare(cind era'acceptat'). Aceste lucruri indica faptul ca printr-o permutare P (intr-o alocare ce accepta sistemul) singurul task care poate fi "defectat" (executia lui nu se poate efectua in perioada proprie) este cel ce trece de la o prioritate mai mare la una mai mica. Fie Tp(k+1) si Tm(k) task-urile asupra carora se aplica permutarea P. Aceste task-uri au perioadele Ip si Im aflate in relatia Ip > Im si prioritatile k+1,respectiv k.

Presupunem, prin absurd ca in urma permutarii P,executia lui Tp nu e satisfacuta in intervalul [t',t'+Ip], capetele acestui interval constituind doua cereri consecutive iar t'+Ip (conform ipotezei di=Ii) reprezinta momentul pina cind poate fi satisfacuta cererea din momentul t'.De-a lungul intervalului [t',t'+Ip] task-urile Tp si Tm au la dispozitie acelasi 'spatiu neocupat',atit inainte cit si dupa efectuarea permutarii (pastrind aceleasi conditii initiale). Prezentam un exemplu in figura 1:

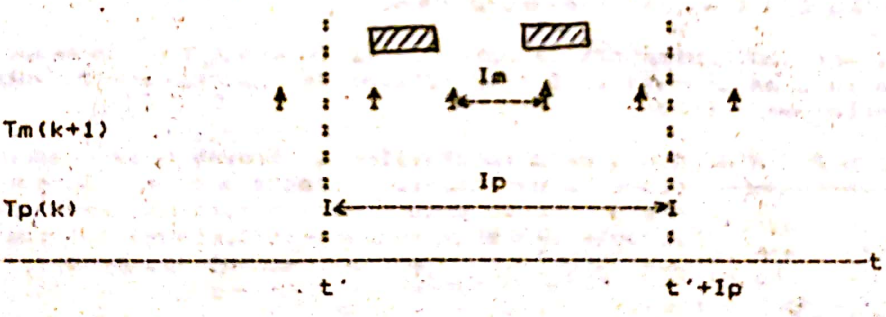


fig.1

↑ cereri de executie

'spatiu neocupat'

Ti(j) task-ul Ti are prioritatea j.

Task-ul Tp(k) poate fi defectat doar de Tm(k+1) care ii 'consuma'(datorita trecerii pe nivel de prioritate mai mare) din 'spatiul neocupat' avut la dispozitie inainte de permutarea P. Sint posibile doua alternative:

a) Pe intervalul $[t',t'+Ip]$ sint executate portiuni din $Tm(k+1)$, corespunzind la cel putin doua cereri ale acestuia. Fie t'' timpul pina la care trebuie satisfacuta executia ultimei portiuni din Tm (t'' corespunde cu o cerere de executie a lui Tp care nu trebuie terminata in intervalul $[t',t'+Ip]$). Atunci:

(4) $t''-t'-D-(r'+r'')*\tau m < \tau p$

D reprezinta timpul ocupat de task-urile cu prioritati mai mari decit k+1.

$r' \in N$ si reprezinta numarul de perioade ale task-ului Tm, cuprinse in intregime in intervalul $[t',t'']$.

$r'' \in R$ si reprezinta fractiunea din m ramasa neexecutata de la o cerere anterioara momentului t', dar cu timpul limita in intervalul $[t',t'']$.

Dar, in aceleasi conditii initiale, inainte de permutare, cu $Tp(k+1)$ si $Tm(k)$

(5) $(t''-t')-\tau p-D > (r'+q'')*\tau m.$

Deoarece $q'' > r''$ (daca cererea lui $Tm(k+1)$ nu a fost satisfacuta inainte de t', cu atit mai putin va fi tratata pentru $Tm(k)$), relatia (4) contrazice relatia (5). Inseamna ca in conditia a) presupunerea ca permutarea P nu conserva 'acceptarea' este falsa.

b) Pe intervalul $[t',t'']$ sint executate portiuni corespunzind unei singure perioade din task-ul Tm, in acest interval fiind prezenta o cerere, dar timpul limita corespunzator este $t'''>t''$.

(6) $Ip-r*\tau m-D < \tau p$

$r<1$ si reprezinta portiunea din m ce se desfasoara in $[t',t'+Ip]$.

Dar, inainte de permutare, deci cu $Tm(k)$ si $Tp(k+1)$, in tom ales astfel incit $t'''=t'+Ip$ (sistemul este acceptat oricare ar fi conditiile initiale), este indeplinit conditia:

(7) $Ip-\tau p-D > \tau m \quad = \quad Ip-\tau m-D > \tau p.$

Deoarece (6) contrazice (7) inseamna ca si in acest caz presupunerea ca in urma permutarii P proprietatea de 'acceptare' nu este indeplinita, este falsa.

(8) **TEOREMA.** Alocarea prioritatilor in ordine nedescrescatoare a perioadelor Ii este optima (Optimul are urmatorul sens: daca o alocare oarecare accepta sistemul atunci si alocarea in ordinea specificata mai sus accepta sistemul)

Demonstratie

Conform acestei teoreme, task-ul cu perioada minima va avea prioritatea maxima. Fiind data o asignare oarecare a permutarilor, asignare ce accepta sistemul, se aplica succesiv permutari de tip P pina se obtine alocarea in ordine nedescrescatoare a perioadelor Ii. Conform lemei (3), permutarile de tip P conserva acceptarea sistemului.

3. CONCLUZII

Am demonstrat ca alocarea prioritatilor in ordine nedescrescatoare este optima. Dar nu este suficient. Pentru a verifica acceptarea sistemului este propus un algoritm pentru task-uri nepreemtibile,in conditii initiale t01=t02=...=t0n=0.

Se noteaza:

I^T:vectorul perioadelor Ii ; $I^T=(I1,I2,\ldots,In)$

τ^T:vectorul duratelor τi ; $\tau^T=(\tau_1,\ldots\tau_\rho)$

x:vectorul de stare al task-urilor x=(x1,x2,...,xn)
xi=1 : procesul i este activ (asteapta executia)
xi=0 : procesul i este inactiv (a fost executat si nu e cerut inca executia).
kan,kn :variabile de lucru.

Cu aceste notatii se construieste algoritmul care va raspunde prin "da" sau "nu" la intrebarea : alocarea in ordine nedescrescatoare a prioritatilor accepta sistemul?

T

1. kan=0;x $=(1,1,..,1)$;accept='yes';t=0
repeat

2. $j=\sum_{i=1}^{n} xi*i* \prod_{k\neq i}^{n} \bar{x}k$

(9)

3. $t=t+\tau j$
pentru i=1,n

4. $ki=\left[\frac{t_i}{T_i}\right]$

if ki>kai then
if xi=1 then
accept ='no'
exit
else
kai=k
xi=1

until j=1
write accept.

Pentru simplificarea notatiilor s-a presupus ca task-ul Tn are prioritatea n (cea mai mare) , task-ul Tn-1 are prioritatea n-1, etc. Algoritmul se termina atunci cind la unul din task-uri s-a depistat o "depasire" (a aparut o cerere de tratare fara ca cea anterioara sa fi fost tratata),sau atunci cind task-ul 1 a fost executat,iesind astfel din situatia cea mai critica (cind e ultimul dintr-o coada de lungime maxima).

Teorema (8) si algoritmul (9) ofera doua instrumente puternice pentru pentru alocarea prioritatilor task-urilor si/sau intreruperilor. Recomandarile diversilor autorilor de a aloca prioritatile in ordinea importantei task-urilor se dovedesc a fi lipsite de continut,caci nu importanta este cea care determina prioritatea,ci perioada Ii.Unui task foarte important i se poate aloca

Scanned with OKEN Scanner

prioritatea minima, cu conditia ca el sa aiba perioada Ii cea mai mare.Desigur ca de la bun inceput am considerat ca tratam cazul unui sistem care nu are voie sa lase evenimente netratate.

Importanta teoremei(8) este foarte mare,caci ea simplifica obtinerea unui raspuns la intrebarea : Exista o alocare a prioritatilor care sa accepte sistemul ? Prin teorema (8) raspunsul la intrebare se da prin studierea alocarii in ordine nedescrescatoare a perioadelor Ti.Fara aceasta teorema, realizarea unui program care sa dea un raspuns acestei intrebari prin evaluarea fiecarei asignari posibile,ar implica foarte mult timp de calcul (daca n=10 ar trebui studiate 10! alocari).
Deasemenea,in cazul utilizarii unui algoritm cu ramificare si marginit (branch and bound algoritm) ,spatiul de memorie ocupat precum si complexitatea algoritmului fac practic imposibila utilizarea acestuia.

BIBLIOGRAFIE

Coffman E.G.JR, Computer and Job-Shop Scheduling Theory, John Wiley & Sons,1976.

Scanned with OKEN Scanner

prioritatea minima, cu conditia ca el sa aiba perioada Ii cea mai mare.Desigur ca de la bun inceput am considerat ca tratam cazul unui sistem care nu are voie sa lase evenimente netratate.

Importanta teoremei(8) este foarte mare,caci ea simplifica oferirea unui raspuns la intrebarea : Exista o alocare a prioritatilor care sa accepte sistemul ? Prin teorema (8) raspunsul la intrebare se da prin studierea alocarii in ordine nedescrescatoare a perioadelor Ti.Fara aceasta teorema, realizarea unui program care sa dea un raspuns acestei intrebari prin evaluarea fiecarei asignari posibile,ar implica foarte mult timp de calcul (daca n=10 ar trebui studiate 10! alocari).
Deasemenea,in cazul utilizarii unui algoritm cu ramificare si marginit (branch and bound algoritm) ,spatiul de memorie ocupat precum si complexitatea algoritmului fac practic imposibila utilizarea acestuia.

BIBLIOGRAFIE

Coffman E.G.JR, Computer and Job-Shop Scheduling Theory, John Wiley & Sons,1976.

Scanned with OKEN Scanner

ALEGEREA ELEMENTELOR DE CALCUL IN SISTEMELE DE CONDUCERE A PROCESELOR

as.ing. Marin Litoiu - I.P.Bucuresti
Facultatea Automatica

Rezumat

In lucrare este prezentata o metodologie de alegere a sistemului de calcul inglobat intr-o instalatie de automatizare. Criteriile ce stau la baza acestei alegeri pot fi ponderate astfel incit alegerea sa fie optima, satisfacind toate cerintele impuse de procesul tehnologic si de beneficiar. Deoarece in faza de proiectare o mare parte din cerinte au un pronuntat caracter vag, pentru alegerea optima a sistemului de calcul se foloseste o metoda bazata pe multimi vagi.

1. Metodologii de proiectare

Diversitatea instalatiilor de automatizare face tot mai dificila sarcina proiectantului de a alege instalatia optima pentru un proces dat. In automatica, ca si in alte domenii, proiectantul are la dispozitie o gama foarte larga de mijloace dar foarte putine elemente de sinteza. De aceea apare necesitatea elaborarii unei metodologii de proiectare a instalatiilor de automatizare, metodologie care sa contina elemente ce-i permit proiectantului o evaluare rapida si corecta a performantelor diferitelor variante de elemente de calcul (calculatoare, automate programabile, elemente distribuite, etc). Instalatia tehnologica se presupune dotata cu traductori si elemente de executie.
In figura 1 si in figura 2 sint prezentate doua metodologii propuse pentru generarea unei solutii de automatizare optima.
Prima metodologie presupune generarea tuturor solutiilor posibile pe baza caietului de sarcini si a bazei de date de care dispune proiectantul si apoi aplicarea unui algoritm de alegere a solutiei optime. Se obtine un optim global ceea ce este foarte convenabil. Dar a genera toate solutiile posibile, chiar in prezenta unei baze sarace de elemente de calcul, este adesea foarte greu. Riscul acestei metodologii este ca va genera optim global doar peste un numar limitat de solutii.

A doua metodologie este destinata sa elimine dezavantajul de care am vorbit la prima si face apel la proiectarea structurata a instalatiilor de automatizare.

Proiectarea structurata a instalatiilor de automatizare presupune, in principiu, doua etape importante:
- proiectarea arhitecturii functionale ce descrie in limbaj de scheme bloc functiile instalatiei de automatizare. Pentru

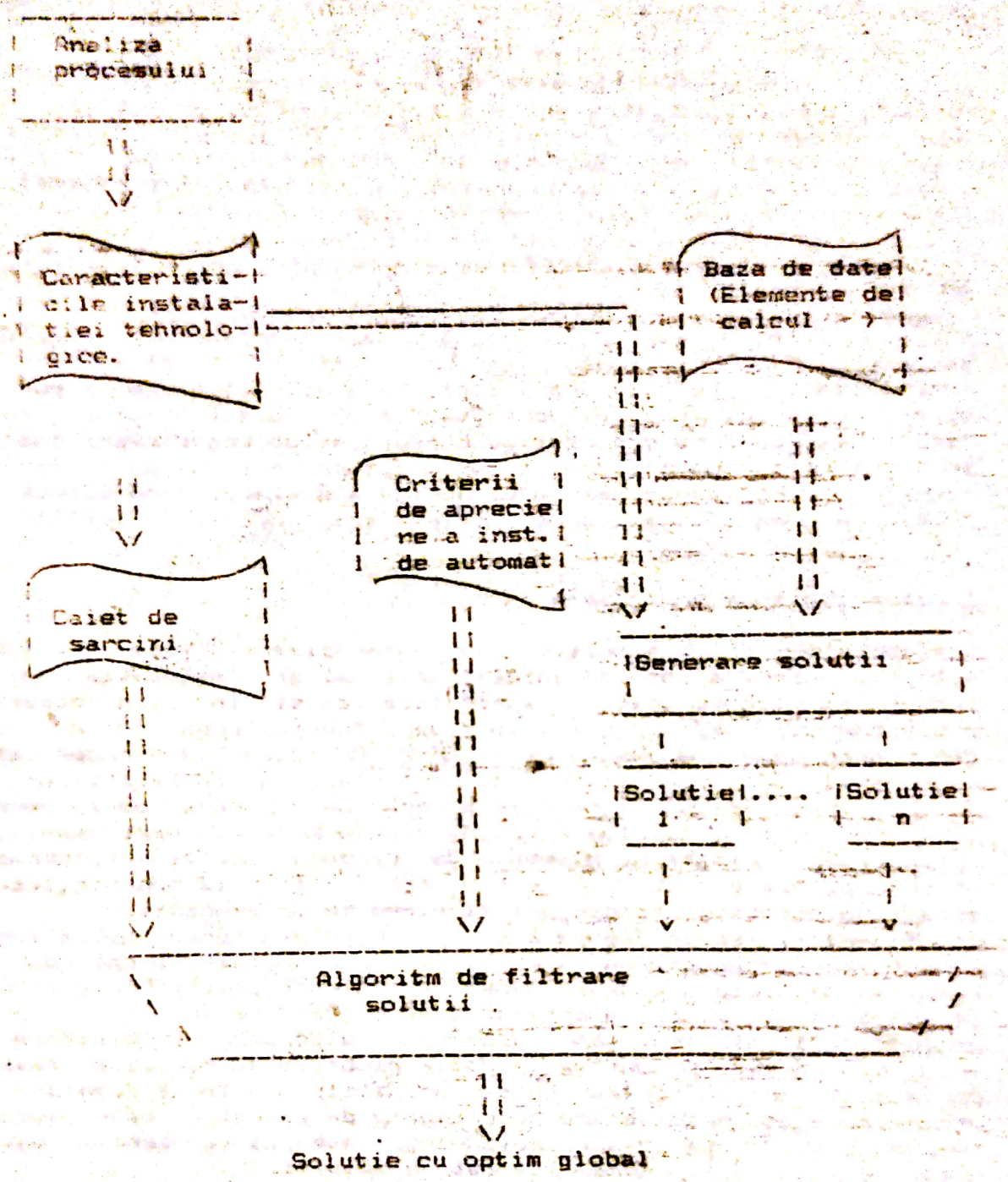


Figura 1. Metodologie pentru configurarea unei instalatii de automatizare cu optim global

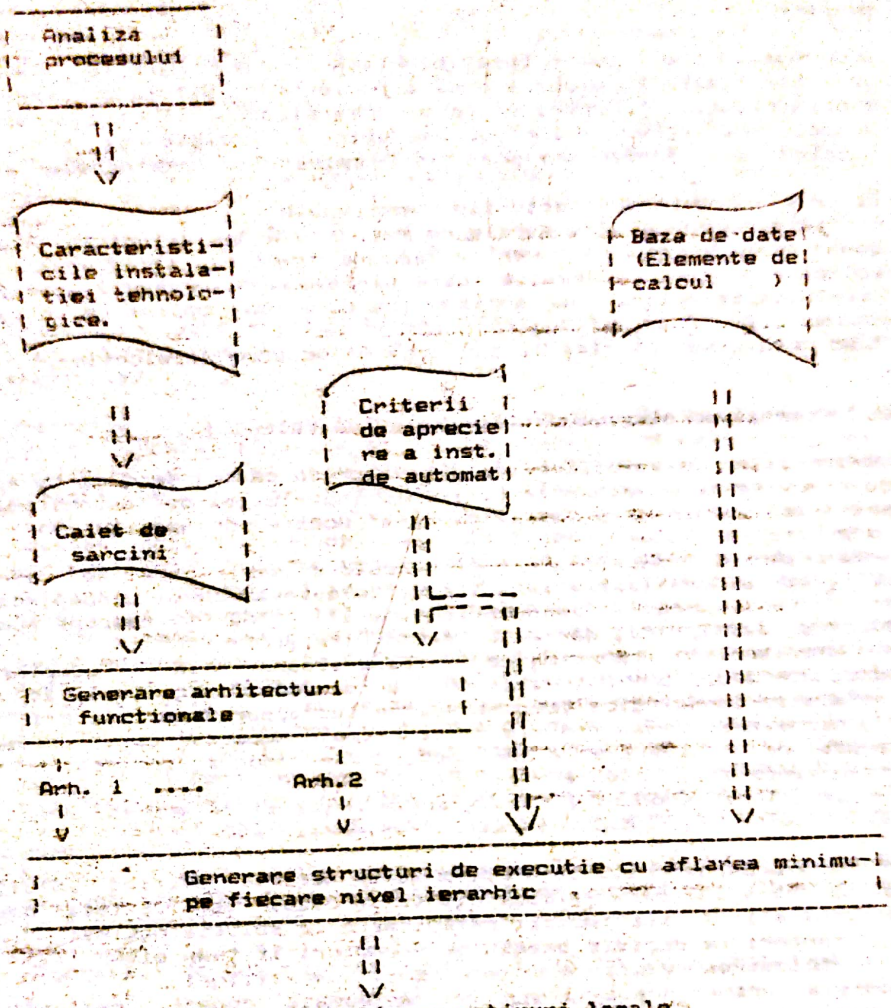


Figura 1. Metodologie pentru configurarea unei instalatii de automatizare cu optimuri locale

instalatiile complexe arhitectura este distribuita si pe mai multe niveluri. De asemenea intotdeauna sint posibile mai multe arhitecturi functionale, in aceasta faza a proiectarii neexistind suficiente date pentru discriminarea celei mai bune solutii.

-proiectarea structurii de executie care materializeaza in limbaj de elemente de automatizare (concrete) arhitecturile functionale .Structurile de executie folosesc elemente din baza

de date .

Deoarece pentru fiecare functie exista un numar mare de elemente de automatizare in etapa a doua a proiectarii exista o 'explozie combinationala '.Pentru evitarea acestei situatii este indicata proiectarea structurilor de executie cu selectarea optimului local.Aceasta proiectare presupune respectarea urmatoarelor etape:

1. se elaboreaza arhitecturile functionale
2. se elaboreaza structurile de executie dupa principiul "top-down" sau "bottom-top".Pentru fiecare nivel al structurilor se aplica algoritmi de decizie care tin seama de criterii specifice nivelului.Se retine doar arhitectura care este optima pe nivelul respectiv.
3. Se reia pasul 2. pina la parcurgerea tuturor nivelurilor.

2.Criterii de alegere a elementelor de calcul

Criteriile de apreciere a elementelor de calcul exprima gradul de adecvare al acestora la cerintele instalatiei de automatizare sau la pretentiile beneficiarului.Aceste criterii pot fi impartite in 4 clase:

a:criterii functionale - sint acele criterii care se refera calitativ si cantitativ la sistemul de automatizare, independent de realizarea tehnologica(timpul de raspuns,numar intrari/iesiri,tipul marimilor de intrare,etc);

b:criterii tehnologice -referitoare la adaptarea echipamentului de automatizare la mediul fizic implicat de instalatia tehnologica(temperatura de lucru,perturbatii,etc);

c:criterii operationale - sint acele aspecte care privesc partea de exploatare a instalatiei (extinderi in spatiu si timp, mentenabilitate, etc.);

d:criterii economice - sint acele criterii ce privesc pretul materialelor, al manoperei, al intretinerii, etc;

Pe baza acestor criterii trebuie luate decizii privind solutia optima atit in cazul global cit si in cazul local.Exista doua posibilitati de alua decizii referitoare la solutia optima:

-arbori de decizie bazati pe structuri **if then else**

-folosirea rezultatelor din teoria deciziilor;

Deoarece prima posibilitate nu modeleaza corect realitatea (aprecierea solutiilor referitoare la un criteriu ar trebui facuta prin da sau nu) si este rigida (aparitia unei noi solutii ar modifica intregul arbore de decizie) se prefera a doua posibilitate de decizie.

Conform teoriei deciziei ,pentru rezolvarea problemei de decizie se construieste tabela de evaluare din **figura 3.**

Prin C1,C2,....Cn se inteleg criteriile de apreciere iar prin A1,A2,.....Am se inteleg alternativele (solutiile) care fac obiectul deciziei.Nu toate criteriile sint la fel de importante relativ la aplicatia pentru care se doreste alegerea solutiei optime de acea fiecaruia i se atribuie o pondere wj care poate fi un numar real cuprins intre 0 si 1 sau o multime vaga.

	Ponderi	A1	A2	.....	Am
C1	w1	r11	r12	.....	r1m
C2	w2	r21	r22		r2m
.	.				
.	.				
.	.				
Cn	wn	rn1	rn2	.....	rnm

Figura 3.

Gradul in care solutia Aj satisface criteriul Ci se apreciaza prin numere reale cuprinse inte 0 si 1 sau prin multimi vagi(rate).

Prezentam in continuare un algoritm pentru selectarea solutiei optime .Acest algoritm se aplica atunci cind ratele rij ca si ponderile wi nu mai sint valori reale ci multimi vagi. Situatia aceasta se intilneste in practica atunci cind exista dubii in legatura cu valorile exacte ale ponderilor si ratelor. In aceasta situatie gradul de satisfacere de catre varianta Aj acriteriului Ci este reprezentata prin functia de apartenenta $u_{PRij}(rij)$, unde rij ia valori pe intervalul [0,1], u_{Rij} luind valori ,de asemenea, tot in acelasi interval.
Ca un exemplu ,sa consideram ca nu exista informatii suficiente sau o metoda exacta de determinare a rij ca o valoare numerica. In acest caz se acorda lui rij mai multe valori ,fiecare valoare fiind insotita de alta valoare $-u_{Rij}(rij)$-care exprima gradul de incredere in valoarea rij.Se poate obtine, de exemplu, multimea rij=(0.8/0.7, 0.6/1) care exprima faptul ca $u_{Rij}(0.8)=0.7$ si $u_{Rij}(0.6)=1$.Deci se are mai mare incredere in valoarea 0.6 alui rij.Pentru o cit mai corecta reflectare a realitatii este indicat ca in aceste situatii rij si u_{Rij} sa fie atribuite de mai multi specialisti.
Aceeasi discutie este valabila si pentru ponderile wi. Cind nu se poate aprecia cu siguranta ponderea criteriului Ci se recurge la exprimarea ponderii prin multimi vagi.

Algoritmul de determinare a valorii optime este urmatorul:

P1. $u_{zi}(z)=[\bigwedge_{t=1}^{n} u_{wj}(wj)] \wedge [\bigwedge_{k=1}^{n} u_{Rik}(rk)]$

Simbolul $\wedge$ desemneaza operatia de aflare a minimului iar z desemneaza o multime (w1,w2,.....wn,r1,r2....rn).

P2. $u_{Ri}(r) = \sup_{z:g(z)=r} u_{zi}(z) \mid g(z)=$ ..... (4)

P3. $u_{I/R}(i/r_1,..,r_m) = \begin{cases} 1 \text{ daca } ri)=rj \ \forall\, j \in I. \\ 0 \text{ altfel.} \end{cases}$

P4. $u_R(r1...,rm) = u_{Ri}(ri)$

P5. $u_I(i) = \sup_{r_1,r_2,\ldots r_m} u_{I/R}(i/r_1,r_2,\ldots r_m)\, u_R(r_1,\ldots r_m)$.

P6. Varianta optima va fi cea care are cea mai mare valoare pentru u_I.

Observatie Ponderile si ratele care nu sint vagi pot fi transformate in multimi vagi cu functia de apartenenta 1.

Bibliografie

1. Baas & Kwakernack *Rating and ranking multiple aspecte alternative ,Automatica IFAC, 1985
2. Claude Sourisse * Automatisme: choisir les technologies de commande. Le Nouvel Automatisme, mai 1984.

Scanned with OKEN Scanner

SISTEM DE CRIPTARE CU CHEIE PUBLICA PRIN METODA FUNCTIILOR UNISENS DE TIP "RUCSAC"

autori Ing. C. RADU
Institutul Politehnic Bucuresti

1. INTRODUCERE

Dezvoltarea sistemelor de "posta electronica" si "transfer bancar electronic" a determinat aparitia sistemelor de criptare cu cheie publica in retelele de calculatoare, pentru a se asigura protec. informatiilor impotriva unor intrusi precum si a partene.ior de retea neavizati sau neautorizati la acces.

Prin sistemul de criptare va trebui sa asiguram doua proprietati importante ale postei clasice:

1. caracterul privat al postei clasice;
2. posibilitatea semnarii mesajelor;

In aceasta lucrare vom prezenta implementarea metodei "rucsacului" de criptare cu cheie publica, metoda ce are proprietatea ca publicarea cheii de criptare a unui mesaj nu face posibila dezvaluirea cheii corespunzatoare de decriptare decit pentru utilizatorul autorizat, posesor al unei informatii suplimentare secrete, numita trapa.

2. FUNCTII UNISENS CU TRAPA. PROPRIETATILE SISTEMELOR DE CRIPTARE CU CHEIE PUBLICA

Ideea criptografierii cu chei publice este strins legata de ideea functiilor unisens cu trapa. Astfel, dindu-se un argument x este usor de calculat valoarea functiei f(x) in timp ce calculul lui x din f(x) este o problema intractabila (nerezolvabila in timp polinomial). Situatia este prezentata in fig.1 :

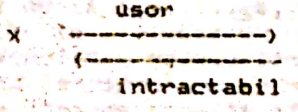


fig. 1

Ne-am referit la f(x) ca la o functie, dar ea trebuie inteleasa in sensul mai larg al unei metode de criptare nondeterministe. Mai mult, calculul lui x din f(x) este o problema intractabila doar pentru un criptanalist neautorizat, dar trebuie sa fie simpla pentru receptorul legal, care este

posesia unei anumite informatii private denumita in mod sugestiv trapa.

Conceptul prezentat anterior face posibil ca intr-un sistem de criptare cu cheie publica, fiecare utilizator sa-si plaseze procedura de criptare E intr-un fisier accesibil tuturor, adica acesta se prezinta ca un director ce contine numele utilizatorului si cheia folosita de el. Dar utilizatorul pastreaza secretul asupra procedurii de decriptare D corespunzatoare. Procedura E apartine clasei functiilor unisens cu trapa si are urmatoarele proprietati :

a) Decriptarea formei criptate a unui mesaj M conduce la obtinerea mesajului M. Formal :

$$D(E(M))=M ;$$

b) Atit D cit si E sint proceduri usor de calculat;

c) Prin relevarea publica a metodei de criptare E utilizatorul nu dezvaluie o cale usor de urmat in vederea calculului lui D .

d) Daca un mesaj M este mai intii decriptat si apoi criptat, M este rezultatul obtinut. Formal:

$$E(D(M))=M ;$$

Cind utilizatorul dezvaluie metoda de criptare, el furnizeaza o metoda foarte ineficienta de calcul al lui D(C) (unde C este mesajul de criptat): testarea tuturor mesajelor M pina cind este gasit acela astfel incit E(M)=C. Daca proprietatea c) este satisfacuta numarul mesajelor ce trebuie astfel testate va fi atit de mare incit timpul de calcul afectat face problema intractabila.

Proprietatea d) este necesara doar pentru implementarea semnaturilor. Orice mesaj este textul criptat al unui alt mesaj si orice text criptat este el insusi un mesaj permis.

Metoda rucsacului de criptare se bazeaza pe urmatoarea problema: se dau un n-uplu: $A=(a_1,a_2,\ldots,a_n)$ si un intreg pozitiv k. Intuitiv k indica marimea rucsacului si fiecare dintre numerele a_i reprezinta marimea articolelor particulare ce pot fi impachetate in rucsac. Este clar ca cunoasterea articolelor ce intra in rucsac conduce imediat la aflarea marimii sale k in timp ce cunoasterea marimii rucsacului nu permite afirmarea articolelor ce il compun. Din acest motiv metoda rucsacului poate fi privita ca o functie unisens cu trapa, trapa informationala fiind reprezentata de cunoasterea setului de articole ce pot printr-un subset sa completeze rucsacul.

Scanned with OKEN Scanner

3. CRIPTAREA/DECRIPTAREA PRIN METODA "RUCSACULUI".

Vom incepe cu enumerarea unui set de concepte utilizate.
Un vector "rucsac" este un n-uplu ordonat de intregi pozitivi distincti a_i astfel $A=(a_1,a_2,\ldots,a_n)$.

O exemplificare a problemei "rucsacului" este perechea (A,k) unde A este un vector "rucsac" iar k este un intreg pozitiv (cu sensul de "continut" al rucsacului).

O solutie pentru (A,k) este un subset al lui A, ale carui elemente sumate dau k. Deoarece vorbim despre un subset fiecare a_i apare in suma cel mult odata. De aceea problema "rucsacului" este uneori denumita si problema sumei subsetului.

Cea mai comuna varianta a problemei este de a spune despre o pereche (A,k) daca are sau nu solutie. O varianta folosita in criptografie este de a produce o solutie pentru o pereche (A,k) data, cind se cunoaste existenta solutiei.

Criptarea consta in folosirea unui vector "rucsac" A pentru a cripta un bloc C de n biti, prin sumarea componentelor din A ce apar in pozitiile corespunzatoare lui 1 din C. Daca suma este notata prin k, atunci decriptarea revine la gasirea lui C din k, sau a gasirii lui C din A si k. Daca avem criptosisteme cu cheie publica. Echivalent daca pe C il vedem ca un vector coloana de n biti, atunci criptarea consta in calculul produsului $k=A*C^T$. Ca o exemplificare sa luam $n=6$ si $A=(3,41,5,1,21,10)$. Atunci mesajul $C=(1,1,0,0,1,0)$ este criptat astfel:

$$k=(3,41,5,1,21,10)*(1,1,0,0,1,0)^T=65$$

Pentru acest vector "rucsac" A, toate criptotextele k sint numere ≤ 81, corespunzind unui $C=(1,1,1,1,1,1)$.

Deoarece unicitate decriptarii este de dorit, vectorii rucsac A ar trebui sa aibe proprietatea ca pentru orice k, toate perechile (A,k) poseda cel mult o solutie. Astfel de vectori "rucsac" A sint referiti ca injectivi. Terminologia este foarte naturala, deoarece injectivitatea lui A determina ca si functia indusa de A sa fie injectiva.

Pentru anumiti vectori A, toate perechile (A,k) sint usor de rezolvat. Vom vedea ca vectorii super-crescatori poseda o astfel de proprietate.

Un criptosistem cu cheie publica se poate baza pe astfel de vectori intr-o maniera clara: Ambii parteneri, emitatorul si receptorul cunosc vectorul A. Pe de alta parte daca un vector B este publicat ca cheie de criptare, atunci receptorul legal trebuie sa aiba o informatie trapa secreta, pentru transformarea lui B si a criptotextului intr-o problema usoara a "rucsacului". Vom arata cum se poate efectua aceasta utilizind anumit tip de vectori super-crescatori. Construirea va fi data mai detaliat.

Un vector "rucsac" $A=(a_1, a_2, \ldots, a_n)$ este crescator respectiv super crescator daca :

$a_j) a_{j-1}$ respectiv $a_j) a_i$ pentru $j=2,3,..n$

Este evident ca orice vector super-crescator este si crescator. Pentru un vector "rucsac" A dafinim:

$$\max A = \max(a_j \mid j=1,2...n)$$

Fie x un numar pozitiv. Vom nota cu [x] partea intreaga a lui x care se bucura de proprietatea ca este cel mai mare intreg mai mic sau egal cu x. Pentru intregii x si m, cu m)=2 vom nota cu (x,mod m) cel mai mic rest pozitiv al lui x modulo m. Este usor de verificat ca au loc urmatoarele doua relatii:

$$(x, \text{mod } m) = x - [x/m] * m$$

$$x = (x, \text{mod } m) + [x/m] * m$$

Vom defini acum doua variante ale notiunii de multiplicare modulara. Sa consideram un vector "rucsac" A, un intreg m) max A si un intreg pozitiv t (m astfel incit cel mai mare divizor comun al numerelor t si m este 1 (adica t si m sint prime).

Daca $B = (b_1, b_2, \ldots, b_n)$ este un vector astfel incit:

$$b_i = (t*a_i, \text{ mod } m)$$

vom spune ca B rezulta din A prin multiplicare modulara cu respectarea modulului m si a multiplicatorului t, sau mai prescurtat cu respectarea perechii (m,t).

Conditia c.m.m.d.c. $(t,m) = 1$ garanteaza existenta elementului invers $u = t^{-1}$ astfel incit :

$$t*t^{-1} = t*u = 1 \pmod m \text{ cu } 1 <= u <= m$$

Aceasta implica ca si A rezulta din B tot prin multiplicare modulara cu respectarea lui m si u (unde m) max B, deoarece elementele b_i sint reduse modulo m).

Daca in locul conditiei m) max A vom inlocui-o cu conditia mai puternica:

$$m > a_i$$

Vom spune ca B rezulta din A prin multiplicare modulara puternica cu respectarea lui m si t.

Sa observam ca acum nu putem concluziona ca A rezulta din B prin multiplicare modulara puternica cu respectarea lui m si u deoarece inegalitatea:

$$m > b_i$$

nu mai are loc in mod necesar. Desigur insa ca A rezulta din B prin multiplicare modulara cu respectarea lui m si u.

Un realizator de criptosistem alege componentele de proiectare A, t, m, B astfel:

1) A vctor rucsac super-crescator;

2) B vector realizat din A prin multiplicare modulara puternica cu respectare lui m si t, unde m si t au fost alese ca pentru multiplicarea modulara puternica.

Vectorul B este publicat ca cheie de criptare si blocuri C de cite n biti sint trimise spre utilizator ca numere k, obtinute din B, in aceeasi modalitate cu k:

$$k' = B * C^T$$

„Un ascultator neneautorizat trebuie sa rezolve problema (B, k') a "rucsacului". Receptorul autorizat calculeaza $k=u*k' \pmod m$ si rezolva problema (A,k). Mecanismul de functionare este dat de lema:

LEMA: Sa presupunem ca $A = (a_1, a_2, \ldots, a_n)$ este un vector super-crescator si B rezulta din A prin multiplicare modulara puternica cu respectarea lui m si t. Sa presupunem mai departe ca $u=t^{-1} \pmod m$, k este arbitrar si $k' = (u*k \pmod m)$. Atunci au loc urmatoarele rezultate:

1. Problema "rucsacului" (A,k) este rezolvabi in timp liniar. Daca o solutie exista este unica.

2. Problema rucsacului (B, k') are cel mult o solutie.

3. Daca o solutie a lui (B, k') exista, ea o egaleaza pe unica solutie a problemei (A,k).

DEMONSTRATIE

1. Orice problema a rucsacului cu un vector A super-crescator poate fi rezolvata in timp liniar prin citirea in A o singura data de la dreapta la stinga, rezultind o solutie unica.

2.&3. Sa presupunem ca un vector de n biti D este o solutie a problemei (B, k'), adica:

$$k' = B * D^T$$

Atunci avem:

$$k = u*k' \pmod m = u*B*D^T \pmod m =$$

$$= u*(t*A*D^T) \pmod m = t^{-1}*t*(A*D^T) \pmod m =$$

$$= A*D^T \pmod m.$$

Deoarece m excede suma componentelor din A, trebuie sa avem $A*D^T < m$. Deoarece si $k < m$, prin definitia lui k, concluzionam ca $k = A*D^T$. Astfel D egaleaza si solutia unica a lui (A,k). Aceasta demonstreaza 3. Deoarece am plecat cu o solutie arbitrara pentru (B, k) si am aratat ca egaleaza unica solutie alui (A,k) am demonstrat de fapt si pe 2.

In continuare vom ilustra aceasta metoda de criptare-decriptare pe un vector cu 10 componente pentru a asigura citirea usoara a exemplului.

4. EXEMPLU

Fie un vector "rucsac" A cu n=10 componente de tip super-crescator dat dupa cum urmeaza:

$$A = (103, 107, 211, 430, 863, 1718, 3449, 6907, 13807, 27610)$$

Alegem $m = 55207$) a_i si $t_1 = 25236$ cu proprietatea ca c.m.m.d.c $(t,m) = 1$. Atunci $t^{-1} = u = 1061$. Prin multiplicare modulara puternica vom obtine componentele lui B astfel:

$103*25236 \pmod{55207} = 103*25236 - [103*25236/55207]*55207 = 4579$

$107*25236 \pmod{55207} = 107*25236 - [103*25236/55207]*55207 = 50316$

.
.
.

vectorul B devenind:

$$B=(4579, 50316, 24924, 30908, 27110, 17953, 32732, 16553, 22075, 53620) = (t*a_i, \text{mod } m)$$

Vectorul B este cheia publica pentru criptare in timp ce A, t, u, m constituie trapa secreta. Cunoasterea lui m sau t sau u faciliteaza aflarea imediata a celorlalte articole, insemnind practic dezvaluirea secretului.

Sa folosim acum cheia publica B pentru cifrarea textului clar: **BUN VENIT LA IASI** . Pentru aceasta vom adopta o codare numerica in care blank-ul are codul 00000, iar literele A-Z valori intre 1-26. Aceste valori vor fi codificate pe 5 biti (32 de cuvinte sint acoperitoare). Astfel:

blank 00000

A	00001
B	00010
.	
.	
.	
Z	11010

Deoarece B poate fi folosit pentru criptarea blocurilor de 10 biti, textul clar trebuie sa fie impartit in blocuri constind din 2 caractere fiecare. In acest mod criptotextul devine:

text clar	codul binar	criptotextul
BU	0001010101	119034
N	0111000000	106148
VE	1011000101	130584
NI	0111001001	192500
T	1010000000	29503
LA	0110000001	128860
I	0000001001	86352
AS	0000110011	120758
I	0100100000	77426

Criptotextul pentru BU a fost obtinut din inmultirea:
$A*(0001010101)^T=30908+17953+16553+53620=119034$
iar pentru celelalte grupuri de litere s-a procedat similar.

Decriptam acum primul numar 119034. Sa notam ca se efectueaza, mai intii o multiplicare modulara cu $u=t^{-1}$:
$k=u*k \pmod m=1061*119034 \pmod{55207}=36665$.
Vom considera acum problema rucsacului (A,36665). Solutia este obtinuta prin scanarea lui A o singura data de la dreapta la stinga. Oricind numarul avut la dispozitie este cel putin egal cu componenta scanata din A, vom pozitiona bitul pe 1 si noul numar este obtinut prin scaderea componentei vectorului A din numarul avut la dispozitie. Altfel pozitionam bitul pe 0 si numarul avut la dispozitie ramine nealterat. Rezultatul poate fi exprimat dupa cum urmeaza:

numar	componenta A	bit
36665	27610	1--
9055	13807	0 |
9055	5907	1 |->U
2148	3449	0 |
2148	1718	1--
430	863	0--
430	430	1 |
0	211	0 |->B
0	107	0 |
0	103	0--

Vectorul de biti original, din care rezulta textul clar BU, poate fi citit de pe ultima coloana de la baza spre virf. Prin decriptarea similara a celorlalte numere vom obtine si restul textului clar.

5. BIBLIOGRAFIE

1. Merkle, R., Secure Communication over Insecure Channels, Commun. of ACM, apr., 1978.
2. Merkle, R; Hellman, M, Hiding Information and Signatures in Trapdoor Knapsacks, I.E.E.E. Trans. on Information Theory, sept. 1978.
3. Arto Salomaa, Public-Key Cryptography, 1990.
4. D.E. Denning. Cryptography and Data Security. Addison-Wesley, 1982.
5. J. Sebery, J. Pieprzyk. Cryptography: An Introduction to Computer Security. Prentice Hall, New York, 1989.

Scanned with OKEN Scanner

OPERATII GEOMETRICE ASUPRA SOLIDELOR MODELATE PRIN OCTREES

Gorgan Dorian, Miron Aurelian, Rusu Dan
Institutul Politehnic Cluj-Napoca

1. Introducere

Diverse domenii de aplicatie necesita reprezentarea geometrica a obiectelor, fiind una dintre cele mai sugestive metode de dialog intre calculator si operator. Operatiile geometrice asupra corpurilor: generare, translatie, diferenta, intersectie, rotatie, scalare, etc, sint rezultatul operatiilor asupra modelelor acestora in calculator.

Printre modelele geometrice mai des utilizate sint:

- modelul B-rep in care muchiile sint elementele de baza. Acest tip de model implica descompunerea solidului intr-un set de fete de ecuatii cunoscute si o posibilitate de conectare a acestor fete, in general printr-un graf;
- modelul C.S.G. (Constructive Solid Geometry). Solidele sint reprezentate prin descompunerea in solide considerate ca primitive grafice (piramida, cub, sfera, paralelipiped, cilindru), care se combina folosind operatori booleeni pentru construirea obiectului dorit;
- Sistemele de modelare oct-trees sint cele mai potrivite modele pentru metoda divide_and_conquer in spatiu, fata de quad_trees in plan.

In aceasta lucrare s-a ales ultimul model precizat, datorita posibilitatilor de prelucrare a corpurilor (prin operatii booleene), si de efectuare a operatiilor asupra codului ce descrie obiectul.

2. Sisteme de modelare oct-trees

Sistemele cu acest tip de modelare sint larg utilizate in proiectarea asistata, recunoasterea formelor, robotica, etc.

Pentru a creea modelul scenei tridimensionale, se porneste de la un cub acoperitor pe care il spargem in 8 octanti (Fig. 1).

Un obiect se aproximeaza prin divizare in cuburi mai mici, pina la o dimensiune minima data. Astfel se pot modela obiecte de complexitati diferite. Numerotarea octanzilor este arbitrara, dar aceeasi in tot modelul (Fig. 2).

Pentru exemplul din figura 2 se va obtine arborele din figura 3 in care s-au folosit notatii:

- plin: tot octantul este umplut cu corpul considerat. In arbore octantul este reprezentat printr-o frunza plina;
- partial: octantul contine parti din corpul considerat;
- gol: nu contine parti din obiectul respectiv. In arbore este reprezentat printr-o frunza goala.

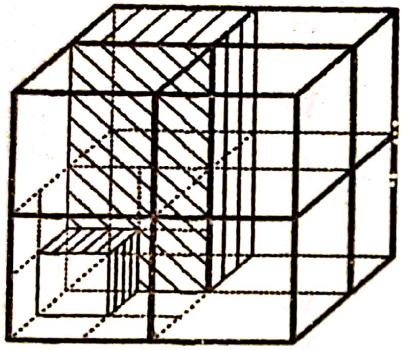


Fig. 1

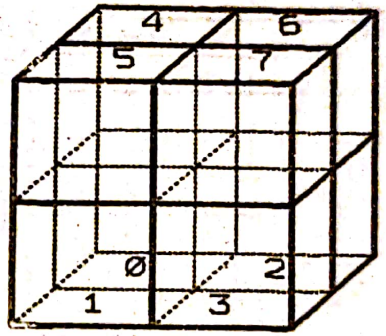


Fig. 2

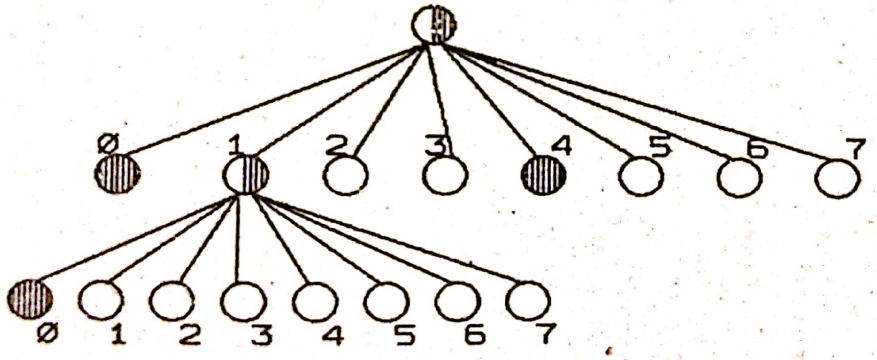


Fig. 3

Orice nod p va fi in continuare divizat, octuplul rezultant fiind formal reprezentat prin arbore.Procesul poate fi continuat pina se ajunge la suprafata ce margineste obiectul. Divizarea se opreste la un nod de dimensiune formal indivizibila. Pentru fiecare nod din arbore se pastreza urmatoarele informatii:

- xmin, ymin, zmin, xmax, ymax, zmax : coordonatele minime si maxime ale cubului nod;
- frunza : un indicator de tip al nodului, cu valoarea true pentru nod frunza si cu valoarea false in alt caz;
- plin : un indicator de nod plin, avind valoarea true pentru nod plin si false in alt caz.

3.Operatii asupra arborilor Oct

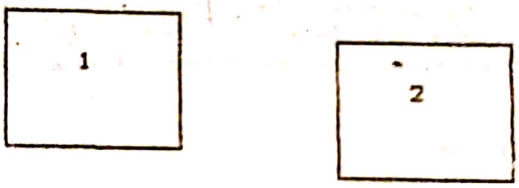
3.1. Diferenta a doua corpuri

Simplificam problema si o reprezentam in plan, cazurile in spatiu fiind aceleasi. Analizam pozitia a doua cuburi frunza pline apartinind fiecare unuia dintre arborii implicati in operatia de diferenta. Analiza o facem pentru toate combinatiile posibile - pentru fiecare frunza plina din primul arbore, parcurgem al doilea arbore si analizam pozitia sa relativa.

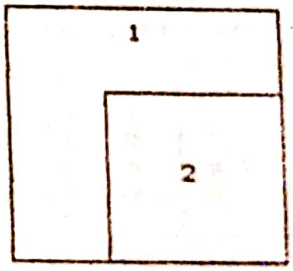
Pot exista urmatoarele cazuri:

notam cu 1 cubul descazut si cu 2 cubul scazator.

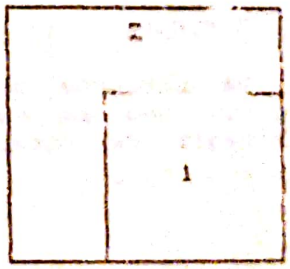
a).cuburi disjuncte.Nu se executa nici o operatie.



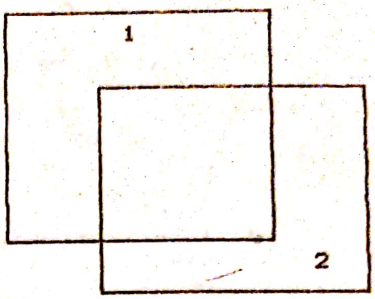
b).cubul descazut in int ul cubului scazator.Cubul descazut e eliminat din arborele rezultant.



c).cubul scazator in interiorul cubului descazut.Se va sparge cubul descazut si se reia prelucrarea in cadrul arborelui descazut.



d).cuburile se suprapun partial. Se sparg ambele cuburi si se reia prelucrarea din nivelul curent al arborelui descazut.



La cazurile c) si d) prelucrarea se opreste la atingerea unei laturi minime:
- daca muchia cubului descazut este mai mica decit latura minima (sau egala) nu se mai sparge si nu se reia prelucrarea;
- daca muchia cubului scazator este mai mica sau egala cu latura minima se sparge numai cubul descazut si se reia prlucrarea (la cazul d).

3.2.Intersectia a doua corpuri

Intersectia presupune aceleasi operatii ca si diferenta, dar actiunile de la cazurile a) si b) difera:
- in cazul cuburilor disjuncte, primul cub (care anterior era descazutul) devine frunză goala in cadrul arborelui rezultant.
- in cazul in care primul cub este in interiorul cubului al doilea, sau egale, primul cub devine frunza plina in cadrul arborelui rezultant si se trece la urmatoarea frunza. plina din primul arbore.

3.3.Translatia unui corp

Se parcurge tot arborele si pentru fiecare nod al sau, terminal sau nu, se aplica operatorul matriceal de translatie, in coordonate omogene:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

$$P' = P * T$$

$$[\, x'\ y'\ z'\ 1\,] = [\, x\ y\ z\ 1\,] * T$$

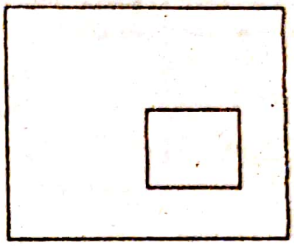
3.4.Rotatia in jurul axelor a unui corp

Se parcurge arborele si pentru fiecare cub ce exista in fiecare nod se reactualizeaza dimensiunile cu operatorii matriceali de rotatie, in coordonate omogene: Rx(u), Ry(u), Rz(u).

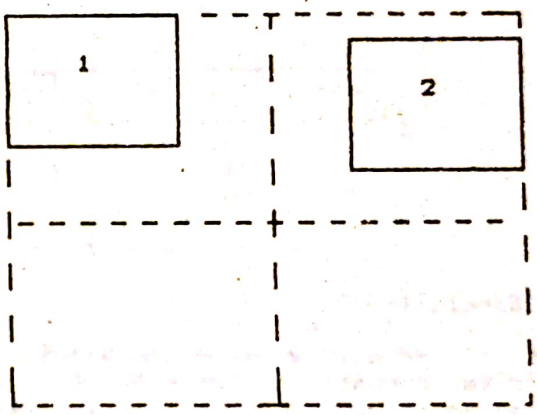
3.5.Reuniunea a doua corpuri

Reuniunea se reduce, in cele din urma, la o intersectie.

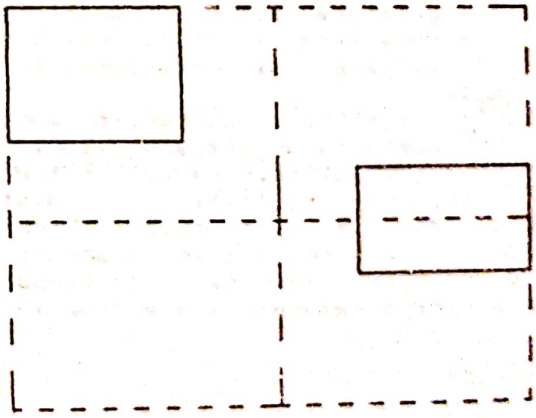
a).In acest caz cubul acoperitor este tocmai unul din cuburile initiale (cel mai mare) si tot el este si arborele rezultant.



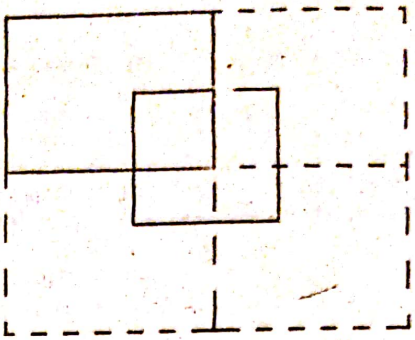
b).Se porneste de la un singur cub acoperitor. Pentru fiecare frunza a sa, testam ce cub acoperitor (din cele doua initiale) contine cuburi din cele doua corpuri. Daca nu contine frunza e goala. Daca contine, frunza este egala cu intersectia ei (considerata plina) si



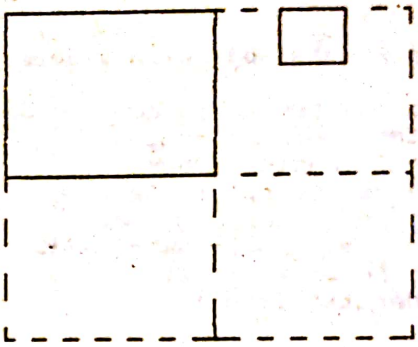
c). operatii la fel ca la punctul b).



d).se porneste de la un cub acoperitor a carui frunza sa fie egala cu cel mai mare dintre cuburile initiale. Aceasta frunza va fi plina. Toate celelalte frunze vor fi intersectiile cu cel de al doilea cub initial.



e).aceleasi operatii ca la punctul d).



4.Concluzii

Programul a fost implementat in limbajul Turbo C, pe un sistem compatibil IBM - AT 286, cu adaptor grafic de tip EGA. Programul este utilizat prin tehnica meniurilor ierarhice, insotite de indicatii (help-uri) de utilizare. Realizeaza functii ca:

- generarea arborilor octree pentru corpuri convexe;
- generarea si verificarea consistentei scenei de obiecte;
- salvarea si incarcarea in memorie a unor arbori deja creati;
- vizualizarea corpurilor modelate prin arbori;
- operarea asupra arborilor;
- dialogul prin meniuri intre utilizator si program.

Arborii rezultati sint destul de mari pentru corpuri complicate si rezolutie mare, dar nu depasesc memoria disponibila. Imbunatatirile urmarite prin modificarea algoritmilor se refera la micsorarea timpului de executie. S-au obtinut urmatoarele masuratori ale timpului de executie:

Scanned with OKEN Scanner

1. Daca latura cubului minima este fixa si egala cu 20 :

Cub acoperitor	Timp construire	Timp vizualizare
40	-	-
80	3"	4"
120	9"	28"
160	14"	30"
200	44"	10' 7"

Timpii de construire nu deranjeaza, corpul odata construit, putind fi incarcat de pe disc. Timpii care deranjeaza sint cei de vizualizare.

2. Testul s-a facut pentru o piramida triunghiulara dreapta cu virful in origine. Cubul acoperitor este fix cu latura 80.

Latura minima	Timp construire	Timp vizualizare
5	2' 3"	9' 50"
10	12"	28"
20	3"	4"
40	-	-

Din punct de vedere al utilizatorului arborii sint identificati prin nume. Implementarea s-a facut astfel incit la un moment dat pot exista maxim 8 arbori in memorie. Pentru a se opera asupra obiectelor acestea trebuie sa aiba arborii corespunzatori incarcati in memorie.

5.Bibliografie

1. Shi-Kuo Chang, Principles of Pictorial Information System Design.Englewood Cliffs, New Jersey: Prentice-Hall, inc., 1989.
2. Foley, J.; Van Dam, A. Fundamentals of Interactive Computer Graphics, Addison Wesley, 1983.
3. Dogaru, D. Elemente de grafica 3-D, Editura Stiintifica si Enciclopedica, Bucuresti 1988.

Scanned with OKEN Scanner

REPREZENTAREA GEOMETRICĂ A MODELELOR TOPOLOGICE ARBORESCENTE

Autori: Gorgan Dorian, Albert Laszlo, Rusu Dan
Institutul Politehnic Cluj-Napoca

REZUMAT:

Obiectul acestei comunicări este prezentarea metodei de definire a modelelor topologice sub formă arborescenta şi a unei aplicatii a acesteia în cartografierea peşterilor.

Un astfel de model poate fi aplicat şi în alte domenii de activitate, cum ar fi:

- căi ferate
- drumuri
- retele de aprovizionare
- cartografie

Aplicatia prezentată este asfel implementată încît să rezolve şi problemele ce derivă din metodele de efectuare a măsuratorilor în cartografierea peşterilor.

Reprezentarea grafică a hartilor este făcută tridimensional, avantajele acestei metode fiind prezentate în lucrare.

Datele necesare construirii reprezentărilor grafice sînt gestionate cu ajutorul unor programe dBASE.

PREZENTAREA MODELULUI

Prelucrarea informatiilor topologice pe calculator impune reprezentarea lumii reale prin modele relativ complexe. Modelul cel mai general pentru reprezentarea legăturilor dintre obiecte este cel de tip retea. În general, modelul de tip retea se poate transpune într-un model ierarhic arborescent fără pierderea de informatii esentiale referitoare la topologie.

Informatiile sînt asociate atît nodurilor cît şi legăturilor din model.

Legătura poate contine următoarele informatii: capacitatea de transport, directia de circulatie, număr de benzi, debit, tipul de legătura (drum, pod, tunel, etc.), forma legăturii (de exemplu dreaptă sau arc).

Nodurile trebuie să conţină obligatoriu informatii despre formă şi pozitie pentru a putea permite reprezentarea geometrică a modelului topologic. În plus, nodurile mai pot contine informatii referitoare la aspect (culoare, simbol), nume, date economice, obiective existente (muzee, restaurante, puncte turistice), etc.

Scanned with OKEN Scanner

Trebuie remarcat faptul că majoritatea nodurilor nu vor conține informații semnificative, ele fiind introduse din alte rațiuni. De exemplu dacă între două noduri legătura nu este omogenă (trecerea printr-un tunel), este necesară introducerea unor noduri noi, astfel încît toate legăturile să fie omogene. Deoarece informațiile geometrice sînt conținute în noduri, schimbarea direcției legăturii sau detalierea topologiei vor duce de asemenea, la apariția unor noi noduri.

Atributele legăturilor pot fi înglobate în noduri fără a reduce generalitatea modelului. Se asigură astfel o omogenitate mai mare în tratarea problemei. Legăturile vor conține deci doar informații structurale ce reflectă modul de interconectare a nodurilor. Avînd în vedere cele prezentate mai sus, deosebim două tipuri de noduri:

- **Noduri informaționale** (I) care conțin informații necesare aplicării modelului într-un anumit domeniu, precum și informații geometrice și cele care reflectă natura și proprietățile legăturii la nodul anterior.

- **Noduri geometrice** (G) care conțin doar informații geometrice și proprietăți ale legăturii la nodul anterior.

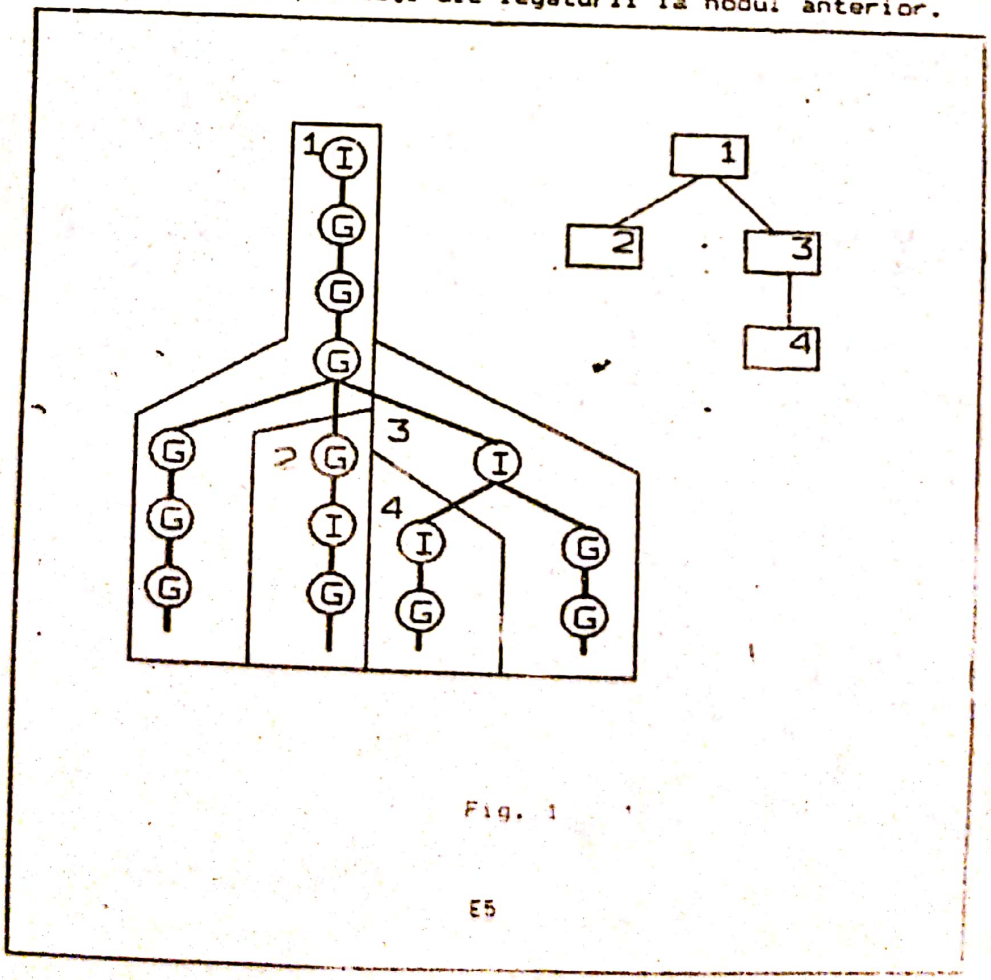


Fig. 1

Urmărind modelul din figura 1 se poate observa existenta unor tronsoane (ramuri) interconectate. In cadrul unui tronson exista atît noduri informaționale cît și geometrice. Oricare dintre aceste noduri poate reprezenta un punct de racordare pentru un nou tronson.

In acest mod de reprezentare a modelului, fiecare tronson devine un șir de noduri. Deși punctele de racordare aparțin tronsonului superior, ele pot caracteriza în același timp tronsonul inferior.

Pentru reprezentarea grafică a topologiei se poate folosi un model bidimensional sau unul tridomensional.

Deși modelul bidimensional are avantajul simplității, noi am optat pentru cel tridimensional, acesta prezentînd o serie de avantaje cum ar fi:

- reprezentarea cît mai fidelă a lumii reale

- posibilitatea reprezentării din diferite puncte de observație;

- generarea unor proiecții pentru construirea de hărți;

- obținerea unor informații de tip "diferență de nivel", inaccesibile în cazul hărților obișnuite.

APLICATIE

Reprezentarea peșterilor este unul dintre domeniile în care modelarea arborescentă a topologiei poate fi aplicată cu succes. Aplicarea acestui model derivă în mod natural din modul în care speologii efectuează cartografierea și măsurătorile într-o peșteră. Măsurătorile pornesc de la intrare și sînt efectuate pe galerii. Galeriile pot fi asociate tronsoanelor prezentate în modelul teoretic.

Informațiile geometrice prezente în noduri vor lua o forma specială ce derivă din modul de efectuare a măsurătorilor. Astfel, pentru fiecare nod măsurătorile se fac în linie dreaptă, relativ la nodul anterior. Se înregistrează următoarele informații referitoare la dreapta care unește cele doua noduri:

a) mărimea (distanța dintre cele două noduri)

b) azimutul (deviația față de direcția NORD)

c) înclinația față de orizontală.

Acest mod se pate asimila cu un model avînd coordonate sferice, relative.

Scanned with OKEN Scanner

O problemă aparte o reprezintă modelarea încăperilor (sălilor) de diverse forme. Culegerea informaţiilor cu privire la lăţimea şi înălţimea galeriilor este foarte dificilă şi ar duce la apariţia a foarte multe noduri datorate legăturilor neomogene.

Rezolvarea acestei probleme se face prin efectuarea unor măsurători suplimentare numite "radiaţii". Dintr-un nod al tronsonului se pot realiza mai multe măsurători radial, pînă la puncte extreme ale sălii, astfel încît circumferinţa acestor radiaţii să reflecte dimensiunile şi forma sălii.

În această reprezentare a modelului fiecare sală devine un şir de noduri geometrice. Legătura va avea ataşată caracteristica "radiaţie" pentru a diferenţia nodurile radiate de cele de tip origine, acestea din urmă participînd efectiv la calculul coordonatelor carteziene absolute ale nodurilor.

Datele referitoare la noduri sînt memorate şi manipulate folosind programe dBASE şi sînt exploatate din Turbo Pascal. Astfel se pot folosi pentru crearea şi întreţinerea fişierelor de date posibilităţile deosebite ale dBASE, iar prelucrarea datelor este făcută în Turbo-Pascal, folosind facilităţile grafice oferite de acest mediu de programare.

Interfaţa cu utilizatorul este realizată pe bază de meniuri. Din meniul principal se poate selecta modul de lucru al programului: afişare arbore de structură (folosit mai ales în verificarea datelor introduse) sau afişarea grafică a hărţii rezultate în urma prelucrării datelor.

Există de asemenea posibilitatea de a încarca, modifica şi salva un fişier de date, toate putînd fi executate în mod de afişare a structurii.

Pentru reprezentarea în mod grafic a hărţilor există posibilitatea de a alege planul de proiecţie sau alegerea unui punct de observaţie pentru o proiecţie ortogonală. Înainte de a putea face reprezentarea grafică a structurii este necesară cunoaşterea sau calcularea coordonatelor carteziene ale punctelor de derivaţie. Dacă în cursul calculării acestor coordonate apar erori acestea sînt semnalate printr-un mesaj ce conţine informaţii despre tipul erorii şi locul (din structură) unde a fost detectată.

CONCLUZII

Deşi aici este prezentat un exemplu de folosire a modelării topologice arborescente, cu un cîmp de utilizare relativ restrîns, principiile care stau la baza acestei lucrări pot fi folosite cu rezultate deosebite şi în realizarea unor hărţi sau diagrame legate de reţele de transport, drumuri, reţele hidrografice, etc.

Scanned with OKEN Scanner

REALISMUL VIZUAL AL REPREZENTARILOR GRAFICE. GRAFTALI

autor: asist. ing. IVAN FARKAS
Univ. Tehnica Timisoara

1. INTRODUCERE

Majoritatea scenelor grafice 3D care urmeaza a fi reprezentate pe calculator pot fi descrise utilizind primitivele grafice asociate unor elemente geometrice ca segmente de dreapta, arce de cerc, poligoane sau curbe de aproximare si de interpolare. Rezultatele obtinute sint de cele mai multe ori multumitoare, atunci cind o aproximare a scenei grafice reale este suficienta. Daca se doreste insa reprezentarea unei scene grafice ce simuleaza aspecte ale mediului natural exterior, de exemplu munti, arbori, nori etc., cu un grad ridicat de realism vizual al reprezentarii , utilizarea primitivelor grafice mentionate este aproape imposibila datorita necesitatii de a defini un numar impresionant de puncte ale proiectiei scenei grafice , precum si relatiile geometrice dintre acestea. Solutionarea acestei probleme in acest cadru devine deosebit de laboricoasa, in mod special atunci cind se doreste reproducerea nu numai a formelor dar si a culorilor si a texturilor.

Reprezentarea unei scene grafice utilizind curbe si poligoane va genera imagini cu un aspect nenatural pentru ca acestea sint prea precise, prea regulate si prea izbutite[1]. Obiectele scenei grafice vor avea un aspect mult mai apropiat de cel natural daca ar putea fi reprezentate cu un grad oarecare dar controlat de iregularitate.

Incorporarea iregularitatilor in reprezentarile grafice implica utilizarea unor tehnici de modelare si transpunere grafica specializate. Intre acestea de o larga raspindire se bucura datorita efectelor realiste, tehnicile bazate pe curbele si suprafetele fractale. Rezultatele generate in aceste conditii sint deseori spectaculoase.

Un fractal poate fi considerat un obiect a carui forma este neregulata la orice scara, indiferent de gradul de detaliere la care este observat, spre deosebire de formele standard ale geometriei euclidiene. Fractalii au o dimensiune fractionala.

Tehnicile de programare utilizind fractalii ofera o cale de generare a unor cantitati enorme de date grafice pornind de la un set relativ redus de reguli de inlocuire. Aceste tehnici pot fi considerate ca un set de operatori care formeaza un asa zis amplificator de baza de date grafica. In acelasi cadru au fost dezvoltate si alte tehnici specifice care isi gasesc aplicatii in reprezentarea multor obiecte din natura. Intre acestea tehnicile bazate pe graftali si sisteme de particule sint deosebit de interesante.[2]

2. GRAFTALI

Notiunea de graftal a fost introdusa de Alvy Ray Smith. Atunci cind in scenele grafice apar obiecte grafice ce modeleaza plante, arbori, arbusti etc., este propice utilizarea graftalilor.

Un obiect modelat ca graftal este construit ca un sir de simboluri de exemplu 0, 1 ,[,]. Semnificatia acestor simboluri este dependenta de tipul obiectului modelat. In cazul unui arbore

semnificatia acordata simbolurilor poate fi urmatoarea:

1 reprezinta un segment de culoare maro a unei ramuri

0 reprezinta un segment de culoare verde a unei ramuri

[reprezinta un punct de ramificatie in arbore si precizeaza inceputul unei noi ramuri

] reprezinta sfirsitul unei ramuri in arbore

Simbolul [este apoi interpretat alternativ ca ramificatie la stinga respectiv la dreapta.

Semnificatia acordata simbolurilor 1 respectiv 0 poate fi modificata, de exemplu pentru a indica ramuri de grosimi diferite sau 1:ramura 0:fruct etc.

Reprezentarea obiectului se poate face simplu cu ajutorul unei proceduri care accepta la intrare un sir de simboluri corect din punct de vedre sintactic, executa comenzile implicate de semnificatia fiecarui simbol si produce imaginea obiectului pe dispozitivul grafic de iesire.

Unghiul cu care se incepe desenarea unei noi ramuri la intilnirea in sirul de intrare a simbolului [poate fi fixat sau poate varia aleator intr-un anumit domeniu de valori. In ultimul caz imaginile obtinute sint mult mai sugestive si diverse.

Actiunile executate la intilnirea simbolurilor 0 respectiv 1 sint in esenta doua:

-se stabileste culoarea de trasare(sau caracteristica codificata)

-se traseaza un segment de dreapta cu o anumita lungime in directia precizata de unghiul curent de trasare. Acest unghi se stabileste in momentul interpretarii simbolului [. Lungimea segmentului poate fi fixata sau poate fi variata aleator intr-un domeniu de valori.

Interpretarea simbolului [conduce la modificarea unghiului de trasare cu valoarea specificata.

Intilnirea in sirul de intrare a simbolului] precizeaza sfirsitul unei ramuri in arbore si implica readucerea pozitiei curente de trasare in punctul in care se afla inaintea ultimului simbol [intilnit. Totodata se readuce unghiul de trasare la valoarea din acel moment.

Actiunile induse de interpretarea simbolurilor [respectiv] necesita retinerea unor parametri de control ai executiei la fiecare nivel de ramificare al arborelui. Fiecare ramura din arbore este la rindul ei un arbore, asemanator intregului, aparinc in pozitii diverse si cu lungimi diferite. Aceasta observatie conduce in mod natural spre utilizarea unei metode de implementare bazata pe tehnica programarii recursive si a primitivelor grafice de iesire in coordonate relative.

3. GENERAREA GRAFTALILOR

Generarea sirului de simboluri se face din aproape in aproape utilizind un set redus de reguli de inlocuire aplicate in mod repetat asupra unui sir initial. Sirul de intrare este parcurs secvential incepind cu primul simbol ignorind parantezele. Fiecare simbol 0 respectiv 1 va genera un nou subsir de simboluri conform cu regulile de inlocuire precizate, iar acesta se va insera in sirul de iesire. Simbolurile [si] se insereaza in sirul rezultat in pozitia curenta, fara nici o modificare.

Un exemplu de reguli de inlocuire este:

0 --> 1[0]1[0]0

1 --> 11

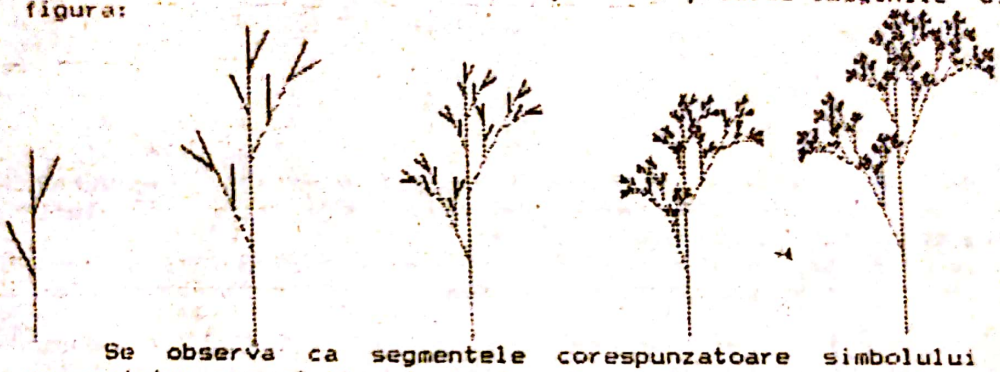
Considerind ca sirul initial de pornire este SI: 0, aplicind in mod repetat regulile de inlocuire vom obtine:

SI1: 0 --> 1[0]1[0]0

SI2: 1[0]1[0]0 --> 11[1[0]1[0]0]1[1[0]1[0]0]1[0]1[0]0

Un numar de aproximativ 7 repetari ale procesului de inlocuire este suficient pentru obtinerea unor imagini sugestive.

Fixind modificarea valorii unghiului de trasare la pi/6 traversarea sirurilor SI1,SI2,...,SI5 vor produce imaginile din figura:



Se observa ca segmentele corespunzatoare simbolului 0 reprezinta ramuri terminale in arbore. In acest exemplu este posibil ca semificatia acordata simbolurilor 0 si 1 sa fie modificata astfel

1 reprezinta ramura

0 precizeaza pozitia de amplasare a unui obiect grafic reprezentind o floare sau o frunza

Imaginea obtinuta in acest caz este desigur mai sugestiva, cu grad sporit de realism vizual.

Calitatea reprezentarii grafice poate fi imbunatatita prin introducerea unui grad de nedeterminism in model. In acest caz fiecare din simbolurile 0 respectiv 1 poate primi o semnificatie multipla. Substituirea simbolulrilor cu una din semnificatiile concrete se va face in mod aleatoriu cu o anumita probabilitate controlata. Avind proceduri de interpretare adecvate se pot obtine imagini deosebit de sugestive.

Marimea obiectului generat depinde de numarul de repetari ale regulilor de inlocuire utilizat in procesul de generare precum si, desigur, de lungimea fizica a segmentelor sau a obiectelor produse ca urmare a interpretarii simbolurilor 0 respectiv 1.

Regulile de inlocuire (1) sint aplicate oridecitoori este intilnit in sirul de intrare unul din simbolurile 0 si/sau 1. Aceasta forma a regulilor de inlocuire se numeste independenta de context.

Un alt exemplu interesant de reguli de inlocuire independente de context este:

0 --> 1

1 --> 1[0][0]1

Pornind de la sirul initial SI:0 ele genereaza nervurile unei frunze, la primul nivel de substitutie.

O alta clasa de graftali se obtine pornind de la un set de reguli de inlocuire dependente de context. In acest caz, subsirul de simboluri care inlocuieste un simbol curent din sirul de intrare depinde de simbolurile precedent si urmator ale simbolului curent. Avind doua simboluri substituibile rezulta un set de opt reguli de inlocuire dependente de context. De exemplu:

```
000 --> 0
001 --> 1
010 --> 0
011 --> 1
100 --> 0
101 --> 00[01]
110 --> 0
111 --> 0
```

Aplicarea acestor reguli de inlocuire necesita precizarea actiunilor in cazurile limita cind codurile de trei simboluri adiacente nu pot fi formate, si anume:

- daca nu exita simbol precedent(la inceputul sirului sau a unei noi ramuri) se va adauga formal un simbol 1 la stinga pozitiei curente
- daca nu exita simbol urmator(la sfisitul sirului sau a unei ramuri) se va adauga formal un simbol 1 la dreapta pozitiei curente

Pornind de la sirul SI:0 se vor obtine sirurile:

```
SI1:   0 --> 00[01]
SI2:   00[01] --> 01[00[01]11]
SI3:   01[00[01]11] --> 00[01]1[01[00[01]11]0]
:
:
```

Traversarea sirurilor obtinute vor produce imaginea unui arbust aplecat in bataia vintului. Inlocuind regula 100 cu 100 --> [01]11 imaginea obtinuta se modifica semnificativ.

4. FRACTALI SI GRAFTALI

Fractalii si graftalii sint obiecte grafice inrudite. Cu toate ca graftalii nu sint fractali deoarece nu contin un grad infinit de detaliere ei sint adeseori numiti subfractali. Setul de reguli de inlocuire care stau la baza generarii graftalilor poate fi extins pentru generarea unor fractali artificiali. Secvente de comenzi grafice in coordonate relative pot fi reprezentate prin siruri compacte construite din simboluri si valori numerice despartite printr-un separator:

T<l> --> traseaza un segment de lungime <l> in directia curenta de trasare

M<u> --> modifica incremental directia curenta de trasare cu valoarea <u>

In acest context regula de inlocuire

T<l> --> T<l/3>;M<60>;T<l/3>;M<-120>;T<l/3>;M<60>;T<l/3>

va genera curbe Kock triadice.

Bibliografie

1. Pokorny,Gerald-Computer Graphics:The Principles Behind The Art and Science
2. Rankin-Computer Graphics Software Construction

REDAREA UMBRELOR IN SCENE GRAFICE 3D MODELATE PRIN RETELE DE POLIGOANE

autori: asist.ing. IVAN FARKAS
ing. GEORGETA FARKAS
Universitatea Tehnica Timisoara

1. Introducere

Realismul vizual al reprezentarilor scenelor grafice in grafica pe calculator este un deziderat, si nu de putine ori o necesitate. In contextul actual al penetrarii masive si sustinute a graficii pe calculator in domenii de o larga diversitate, stacheta exigentei relative la calitatea reprezentarilor este din ce in ce mai ridicata. Reprezentarile grafice, pe linga informatia propriu-zisa pe care o transmit, fie prin ele insele fie completind informatiile text, trebuie sa asigure si un cadru placut, sugestiv pentru receptarea mesajelor. In aceasta directie, realismul vizual al reprezentarilor grafice joaca un rol de prima importanta indiferent daca grafica este utilizata in aplicatii pentru comunicatia cu utilizatorul sau daca este obiectul principal al aplicatiei (animatie,grafica de prezentare,jocuri,etc.)

Obtinerea realismului vizual al reprezentarilor grafice este un proces in general complex, care solicita resurse hardware si software considerabile. Timpul de calcul implicat in acest proces este uneori deosebit de mare ceea ce face ca, atunci cind aplicatia este sensibila din punctul de vedere al timpului de elaborare al imaginilor grafice,acest proces sa fie intrerupt in diverse faze ale sale. In aceste conditii calitatea reprezentarilor este desigur afectata.

2. Reproducerea umbrelor in contextul realismului vizual

Etapele pricipale parcurse in procesul obtinerii realismului vizual corespund unor prelucrari grafice de complexitate din ce in ce mai mare. Vom considera ca obiectele componente ale unei scene grafice sint modelate prin retele poligonale.

Prima etapa cuprinde procesul de indepartare al liniilor si suprafetelor ascunse pentru obiectele scenei grafice 3D. Atunci cind un obiect este observat dintr-un anumit punct in spatiu, anumite fatete ale sale sint vizibile complet, altele ascunse partial sau total. Indepartarea suprafetelor ascunse este o necesitate atit in procesul obtinerii realismului vizual al imaginilor generate cit si pentru reducerea drastica a efortului

de prelucrare solicitat in etapele urmatoare.

Etapa urmatoare cuprinde ansamblul de prelucrari orientate pe redarea iluminarii scenei grafice. Reprezentarea finala a suprafetelor depinde de tipurile surselor de lumina amplasate in scena , de proprietatile fizice ale suprafetelor(culoare, textura, reflectivitate) precum si de pozitiile si orientarile suprafetelor in raport cu sursele de lumina.

O noua etapa calitativ superioara este reproducerea proprietatilor suprafetelor obiectelor. In aceasta etapa sint aplicate tehnici specifice de texturare si nuantare a culorilor, obtinindu-se efecte vizuale deosebite.

Un grad sporit de realism vizual se obtine prin reproducerea umbrelor aruncate de obiectele iluminate asupra celorlalte obiecte din scena grafica.

In cadrul lucrarii se prezinta o posibila implementare intr-un pachet grafic elaborat in limbaj evoluat (TPascal) a metodei de reproducere a umbrelor avind la baza o varianta a tehnicii poligoanelor detaliu asociate, si integrarea acesteia in fluxul prelucrarilor grafice care conduc la reprezentarea scenelor grafice 3D.

3. Tehnica poligoanelor detaliu

Tehnica poligoanelor detaliu se bazeaza pe observatia ca suprafetele care sint in umbra sint ascunse observatorului atunci cind scena este privita din punctul de amplasare al unei surse de lumina. Pe baza acestei observatii, procesul de determinare al umbrelor este similar cu procesul de identificare al suprafetelor ascunse.

Tehica poligoanelor detaliu poate fi aplicata cu succes in mod deosebit in cazul in care modelarea poligonala este combinata cu o metoda de transpunere pe dispozitivul grafic de iesire raster bazata pe sortarea in adincime si pe algoritmul de acoperire painter.

Determinarea umbrelor ca etapa de procesare, poate fi facuta in paralel cu procesul de identificare al suprafetelor ascunse. Algoritmul de identificare al suprafetelor ascunse stabileste care sint suprafetele vizibile din punctul curent de observare, in timp ce algoritmul de determinare a umbrelor stabileste care sint suprafetele "vizibile" din punctele in care sint amplasate sursele de lumina.

Esenta procesului de determinare al umbrelor este observatia ca acele suprafete poligonale care sint vizibile atit din punctul de observare cit si din punctul de amplasare al sursei nu pot fi in umbra iar acelea care sint vizibile din punctul de observare dar nu si din punctul de amplasare al sursei sint in umbra. Aceasta logica poate fi extinsa pentru surse punctuale multiple[1].

In principiu, fiecarui poligon vizibil din punctul de observare care este complet sau partial vizibil din punctul de amplasare al sursei (este "in lumina") i se ataseaza un poligon secundar coplanar numit poligon detaliu. Umbra aruncata de un poligon P pe un poligon Q este un nou poligon U. Pentru un

poligon P se determina reuniunea poligoanelor umbra determinate de celelalte poligoane ale scenei care sint in lumina de la cel putin o sursa. Acest poligon rezultat R este poligonul detaliu asociat poligonului P. Poligonul P se transpune in ecran utilizind iluminarea calculata pentru acesta iar apoi se transpune poligonul R (peste P) luind in calculul de iluminare numai intensitatea luminii ambiante.

4. Implementarea tehnicii poligoanelor detaliu

Vom considera ca in scena grafica sint prezente N surse de lumina punctiforme, o iluminare ambientala si un numar oarecare de corpuri solide descrise prin retele de poligoane nonpenetrante care nu se acopera in mod ciclic.

Ansamblul de poligoane care descriu obiectele prezente in scena grafica este supus procesului de identificare al suprafetelor ascunse in paralel cu procesul de determinare a vizibilitatii din cele N puncte de amplasare ale surselor de lumina. Cele doua procese fiind in esenta aceleasi, se utilizeaza aceeasi procedura bazata pe evaluarea cosinusului unghiului facut de normala la suprafata poligonului(orientat) si directia de observare respectiv directiile date de amplasarea surselor. In structura de date atasata unui poligon sint marcate rezultatele procesarii: daca este ascuns sau nu si numerele de identificare ale surselor fata de care poligonul este in lumina.

In etapa urmatoare poligoanele sint sortate in functie de departarea centrului de grutate al poligonului fata de pozitia de observare. Se obtine o lista avind NRP poligoane sortate care corespunde cu ordinea in care poligoanele vor fi transpuse in ecran. Primul poligon din lista este poligonul cel mai indepartat de pozitia de observare.

Datorita ordonarii poligoanelor, pe un poligon P_i vizibil din punctul de observare pot apare in principiu umbre aruncate de oricare din poligoanele P_j ($i<j<=NRP$)care nu au fost inca transpuse in ecran si care sint in lumina relativ la cel putin una din sursele de lumina. Poligoanele P_j ($j<i$) care se gasesc in spatele poligonului P_i nu sint luate in considerare. Umbrele aruncate de acestea vor apare, eventual, pe fata poligonului P_i care nu este vizibila din punctul de observare.

Dupa transpunerea in ecran a poligonului P_i, se determina cite un poligon detaliu $P_{j,i}$ generat de fiecare poligon P_j, $i<j<=NRP$, care este in lumina fata de cel putin una din cele N surse de lumina. Acest poligon obtinut se transpune in ecran luind in calculul de iluminare doar intensitatea luminii ambientale.

Determinarea unui poligon $P_{j,i}$ (poligon detaliu generat de poligonul P_j relativ la poligonul P_i) presupune urmatoarele operatii:

-se determina ecuatia planului poligonului P_i

-se realizeaza o proiectie paralela a poligonul P_j pe planul poligonului P_i, considerind o iluminare paralela din directia sursei K, obtinind poligonul detaliu $P_{j,i}$

-se detrmina extinderile pe ox respectiv oy ale

poligoanelor P_i si $P_{j,i}$ proiectate in spatiul ecranului si se face un test de suprapunere al extinderilor. Daca cele doua poligoane nu se suprapun se trece la procesarea urmatorului poligon din lista.

-daca poligoanele se suprapun , poligonul $P_{j,i}$ se supune unui proces de clipping pentru poligoane(de exemplu Sutherland-Hodgman), considerind ca regiune de clipping poligonul P_i. Se obtine poligonul sectionat $Q_{j,i}$.

-daca poliponul $Q_{j,i}$ exista, se transpune in ecran luind in calculul de iluminare intensitatea luminii ambientale

4. Concluzii

Metoda prezentata produce reprezentari grafice cu grad ridicat de realism vizual. Este recomandata atunci cind scenele grafice sint de complexitate medie, timpul de executie crescind substantial atunci cind numarul poligoanelor din scena grafica este foarte mare.

Bibliografie

1. Van Damm -Principles of Interactive Computer Graphics
2. Rankin-Computer Graphics Software Construction
3. -Mathematical Elements for Computer Graphics

Scanned with OKEN Scanner

Metoda de recunoastere a obiectelor poligonale

Silviu Ifrim
Facultatea de Automatică şi Calculatoare
Institutul Politehnic Iasi

Rezumat

Această lucrare prezintă o metodă eficientă de recunoaştere a obiectelor poligonale supuse la translaţie, rotaţie şi scalare cu un factor global din imagini alb-negru. Metoda de recunoaştere se bazează pe găsirea corespondenţei între forma analizată şi o bibliotecă de modele anterior creată. Compararea cu modelele din bibliotecă se face în două treceri în fiecare din acestea utilizînd un anumit tip de trăsături.

1. Introducere

Tehnicile de clasificare a formelor bazate pe contur au constituit o importantă zonă de cercetare în ultimii ani şi au cuprins metode statistice, analiza Fourier a unor funcţii dependente de contur, metode sintactice etc. Recunoaşterea unui obiect implică realizarea unei corespondenţe între o apariţie particulară a acestuia şi un model memorat anterior într-o bibliotecă de obiecte cunoscute. Această corespondenţă se realizează prin compararea unor caracteristici. Ordinea în care se determină valorile caracteristicilor în model şi într-o apariţie particulară depinde însă de poziţia în care sînt considerate aceste obiecte. Este necesară deci folosirea unor algoritmi de aliniere a componentelor celor doi vectori sau a unor etape pregătitoare de căutare sau indexare a căror complexitate este mare (în general $O(n^3)$) [2]. Cum dimensiunea acestor vectori şi a numărului de modele este în general mare timpul necesar recunoaşterii creşte simţitor.

În această lucrare pentru recunoaşterea obiectelor poligonale s-au ales două tipuri de trăsături şi anume unghiurile formate de laturile poligonului şi lungimile acestor laturi. Este evident că primul tip este invariant la translaţie, rotaţie şi scalare în timp ce al doilea este invariant la primele două operaţii şi liniar în raport cu cea de a treia [4]. Aceste două tipuri de trăsături asigură totodată un plus de acurateţe în recunoaştere şi se pot calcula uşor. Nici pentru aceste trăsături însă nu se poate stabili un punct de start şi o ordine de calcul invariabilă în raport cu poziţia obiectului. Pentru a rezolva această cerinţă de aliniere s-au folosit în compărare vectorii prelucraţi pe baza unei transformări descrise în continuare.

2.Descriere transformării Rapide

Transformata Rapidă realizează o corespondență între coeficienții unui vector de intrare și un vector de ieșire având proprietatea că vectorul de ieșire este invariant la deplasări ciclice ale componentelor vectorului de intrare (1). Considerăm un vector $V=(a_1,\ldots,a_n)$ unde n este de forma 2^m. Operatorii folosiți în transformare sînt suma (+) și modulul diferenței $|-|$ fiind posibilă eventual și o altă alegere. Definirea transformatei se poate face utilizînd următoarele formule recursive :

(1) $R(a_1,\ldots,a_n) = (T^m(a_1),\ldots,T^m(a_n))$ unde :

(2)
$$\begin{cases} T^0(a_i) = a_i & \forall i = 1,2,\ldots,n \\ T^k(a_{2i-1}) = T^{k-1}(a_i) + T^{k-1}(a_{i+n/2}) & \forall i = 1,2,\ldots,n/2 \\ T^k(a_{2i}) = |T^{k-1}(a_i) - T^{k-1}(a_{i+n/2})| & \forall i = 1,2,\ldots,n/2 \end{cases}$$

Se poate demonstra că transformata astfel definită este liniară și invariantă la deplasări ciclice ale componentelor vectorului de intrare. Calculul ei se face utilizînd operații simple iar complexitatea algoritmului de calcul este $O(n\log n)$. În ipoteza păstrării aceluiași sens de efectuare a operațiilor pentru cele două trăsături aplicarea transformatei vectorului model și a celui de test determină alinierea componentelor.

Un inconvenient acestui procedeu îl constituie faptul că numărul de componente din vector trebuie să fie o putere a lui 2 ceea ce este dificil în aplicațiile practice. O metodă de evitare a acestui dezavantaj folosită și în experimentele efectuate este completarea numărului de componente ale vectorului prin inserarea de zerouri astfel : dacă $2^{m-1} < n < 2^m$ atunci se selectează primele $2^m - n$ componente în ordinea mărimii lor și se inserează un zerou după fiecare din ele.

3.Algoritmul în recunoaștere

Considerăm un poligon oarecare P cu n vîrfuri situat într-un reper cartezian (modul) și poligonul obținut în urma operațiilor de scalare cu un factor de scală global ($x = x'\alpha$, $y = y'\alpha$) și apoi rotit și translat.

Este evident că unghiurile celor două poligoane sînt două cîte două egale iar laturile două cîte două proporționale. Prima etapă a algoritmului creează bibliotecă de obiecte sub forma unei colecții de înregistrări exprimînd

numărul de vîrfuri, vectorul conținînd mărimea unghiurilor și cel care conține mărimea laturilor după aplicarea transformării descrise anterior.

Faza de clasificare propriu-zisă pornește cu un algoritm de depistare și calculare a coordonatelor vîrfurilor bazat pe detecția segmentelor de dreaptă prin metoda transformatei Hough [5] urmată de determinarea celor doi vectori utilizați anterior -unghiuri, laturi-,și a transformatei lor. Căutarea în colecția creeată se face pe baza numărului de vîrfuri și a distanței între vectorii transformați cuprinzînd mărimea unghiurilor. În cazul cînd această distanță se găsește sub un anumit prag se selectează intrarea corespunzătoare din bibliotecă și se determină o medie a coeficientului de proporționalitate laturilor în raport cu vectorul calculat pentru poligonul de recunoscut:

$$(3) \qquad \alpha_{med} = \frac{\sum_{i=1}^{n} T^m(\ell_i')}{\sum_{i=1}^{n} T^m(\ell_i)}$$

unde ℓ_i și ℓ_i' sînt lungimile laturilor celor două poligoane. Se are în vedere că transformata este liniară deci:

$$(4) \qquad T(\alpha * a_i) = \alpha * T(a_i)$$

Se verifică dacă acest coeficient satisface cu o anumită aproximare δ, aleasă convenabil, proporționalitatea fiecărei componente pentru vectorii din bibliotecă selectați în prima fază și se rețin aceia pentru care este satisfăcută relația (5):

$$(5) \qquad \left| \alpha_{med} - \frac{T^m(\ell_i')}{T^m(\ell_i)} \right| < \delta \qquad \forall\, i = 1,2,\ldots,n$$

Se reia operația de comparare considerînd vectorul lungimilor scalat cu coeficientul găsit și decizia finală se ia în funcție de distanța vectorilor transformați corespunzători.

Întrucît în evaluarea acestor distanțe sau folosit vectorii transformați nu este necesară conform proprietății transformatei Rapide păstrarea aceleiași ordini de înregistrare în vector a componentelor sau a unei alinieiri ulterioare.

4.Concluzii

Rutinele algoritmului prezentat au fost implementate și testate pe un calculator compatibil IBM-PC în limbajul Turbo-C folosind imagini alb-negru cu o rezoluție de 256 X 256 puncte. Metoda poate fi extinsă la curbe oarecare avînd în vedere posibilitatea aproximării acestora prin linii poligonale și alegînd o altă trăsătură în locul unghiului dintre segmente avînd în vedere posibilitățile diferite de poziționare a vîrfurilor. O modalitate de a crește robustețea acestui algoritm constă în eliminarea unor vîrfuri corespunzătoare unor unghiuri mari (apropiate de 180°) care pot ascunde erori în faza de preprocesare.

Bibliografie

1.R.A. Al-Mohamad - "3D Shape Classification using the R- transform", 10 th International Conference on Pattern Recognition, IEEE Computer Society Press 1990.

2.Fridtjof Stein, Gerard Medeoni - "Efficient Two Dimensional Object Recognition ", 10 th International Conference on Pattern Recognition, IEEE Computer Society Press 1990.

3.Ilan Schreiber, Moshe Ben-Bassat - "Polygonal Object Recognition", 10 th International Conference on Pattern Recognition, IEEE Computer Society Press 1990.

4.Dorian Dogaru - "Metode noi în proiectare. Elemente de grafică tridimensională", Ed. Științifică și Enciclopedică, București 1988.

5.D.H. Ballard, C.Brown - "Computer Vision", Prentice Hall 1982.

Scanned with OKEN Scanner

SISTEM DE LABORATOR PENTRU DEZVOLTAREA APLICATIILOR DE CONDUCERE IBM-PC SPOT

Autori: Dr.ing. D. Carstoiu
Ing. C. Radu
Ing. M. Litoiu
Institutul Politehnic Bucuresti

1. INTRODUCERE.

Lucrarea de fata propune constructia cu echipamente existente in tara a unei aplicatii de laborator, destinata in principal instruirii in demeniul conducerii proceselor. Structura propusa este compusa dintr-un echipament SPOT, avind ca scop exploatarea particularitatilor sale legate de existenta plachetelor de interfata pentru cuplare la proces, si un calculator IBM PC (sau compatibil) pentru facilitatile legate de viteza de prelucrare cit si avantajelor oferite pentru realizarea consolei operatorului de proces.

Cele doua echipamente sint conectate printr-o legatura seriala ca suport de comunicatie intre echipamentul SPOT si calculatorul IBM PC.

Din punct de vedere software s-au realizat pachetele de programe pentru achizitie de date, prelucrare primara, transmisie comenzi, programele de comunicare scrise in C AZTEC la echipamentul SPOT. La calculatorul IBM PC pe linga programele de comunicatie s-a realizat un pachet de programe de consola in C++ in care s-au folosit concepte ale programarii orientate obiect.

2. ARHITECTURA HARDWARE A SISTEMULUI

Stiut fiind faptul ca un calculator IBM PC nu este realizat in varianta industriala pentru a permite utilizarea sa in medii perturbate prin cimp magnetic si electric, precum si in medii cu praf si agenti corozivi se incearca utilizarea acestuia si in aplicatii de conducere datorita atit a puterii de calcul cit si a gamei mari de programe realizate sub sistenul de operare DOS.

Echipamentul MINISPOT are o interfata seriala de tip SIO, microcalculatoarele din familia IBM PC, XT, AT folosesc pentru realizarea interfetei seriale circuitul 8250 UART (universal asynchronous receiver transmiter). Sistemul de operare DOS suporta doua porturi seriale (recunoscute de sistemul de operare sub numele de COM1 si COM2), si deci doua circuite UART. Adresele de baza ale celor doua porturi sint tinute la 0040:0000 pentru COM1 si la 0040:0002 pentru COM2. Adresa de baza pentru COM1 este 3F8H iar pentru COM2 este 2F8H. Pentru simplificare, referirile se vor face la registrele numerotate 3FxH dar sint valabile si pentru registrle de la 2FxH.

Circuitul 8250 are 10 registre de 8 biti selectate ca sapte porturi cu adrese distincte, 3F8H-3FEH. In cinci cazuri, registrele sint accesate la o adresa particulara de port, in functie de valoarea bitului 7 din registrul de control al liniei. In continuare sint prezentate registrele :

3F8H(OUT ,bit 7=0 la 3FBH)	Registru de transmisie
3F8H(IN ,bit 7=0 la 3FBH)	Registru receptie data
3F8H(OUT ,bit 7=1 la 3FBH)	Baud-rate low
3F9H(OUT ,bit 7=1 la 3FBH)	Baud-rate high
3F9H(OUT ,bit 7=0 la 3FBH)	Registru de activare intrerupere
3FAH(IN)	Registru de identificare a intreruperii
3FBH(OUT)	Registru de control al liniei
3FCH(OUT)	Registru de control al modemului
3FDH(IN)	Registru de stare a liniei
3FEH(IN)	Registru de stare a modemului

3. STRUCTURA SISTEMULUI DE PROGRAME

Sistemul de operare realizeaza initializare prin INT 14H, functia 0. In DX se incarca numarul canalului de comunicatie (0 pentru COM1 si 1 pentru COM2). In AL se incarca octetul de comanda ce contine lungimea cuvintului, numarul bitilor de stop, tipul paritatii si viteza de transfer.

Se da mai jos un exemplu initializare a canalului 1 de comunicatie seriala cu toarele caracteristici: lungimea cuvintului de 8 biti, 2 b de stop, paritate impara, rata de transmisie 1200 bps:

```
    MOV   AL,0                ;sterge al
    OR    AL,00000011B        ;lungime cuvint
    OR    AL,00000000B        ;biti de stop
    OR    AL,00011000B        ;paritate
    OR    AL,10000000B        ;rata de transmisie
    MOV   AH,0                ;functia 0
    MOV   DX,0                ;COM1
    INT   14H
```

Programatorul poate actiona direct asupra registrelor circuitului 8250, functie de semnificatia bitilor registrului de control al liniei. In continuare se prezinta listingul scris in limbaj C al programelor de comunicatie implementate pe IBM PC.

```
/*****************************************
functii de comunicare :
     transmit date
     receptioneaza date
*****************************************/

#include <stdio.h>
#include <string.h>
#include <dos.h>
```

```
#define EOADR 0x32
#define BOADR 0x31
#define SOVAL 0x33
#define NRCAN 24
//structura  pentru convertor
typedef struct{
   unsigned char vallow;       //LSB convertor
   unsigned char valhigh;      //MSB convertor
   char adr_can;               //adresa canale MUX
   char activ;                 // canal ativ/inactiv
}pach;
pach    par_CAN[NRCAN];

void prog_8250()               //programare 8250
{
asm {
        mov al,0
        or  al,11b               //lungime cuvint:8biti
        or  al,100b              //2 biti de stop
        or  al,11000000b         //viteza 4800
        mov ah,0
        mov dx,0         //COM1
        INT 14h
        mov al,0
        or  al,11b               //lungime cuvint:8biti
        or  al,100b      //2 biti de stop
        or  al,11000000b //viteza 4800
        mov ah,0
        mov dx,1
        INT 14h             //COM2
}
}
void trans_char( char c)//scriere un caracter la COM1
{
asm{
        mov ah,1  //functia 1 de la INT 14
        mov dx,0  //COM1
        mov al,c
        INT 14h
  /*    mov stcom1,ah
        mov ah,1  //functia 1 de la INT 14
        mov dx,1  //COM2
        mov al,c
        INT 14h
        mov stcom2,ah */
  }
}

unsigned char rec_char(){//receptie un caracter
unsigned char st_rec1;
unsigned char  car_rec;
st_rec1=0x80;
while (st_rec1 )= 0x80 )
{
        asm{
```


Scanned with OKEN Scanner

```
                mov ah,2
                mov dx,0
                INT 14h
                mov car_rec,al
                mov st_rec1,ah
            }
    }
    return(car_rec);
    }
    void rec_str(char *string){
            char    c[2]="a";
            while(((c[0]=rec_char())!=EOF)&&(strlen(string)<=30))
                strcat(string,c);
    }
    void trans_adr(void){ //transmitere adrese pentru canale active
                          //  adresele  sint delimitate de  BOADR  si
                          //EOADR
            char    i;
            trans_char(BOADR);
            for     (i=0;i<NRCAN;i++)
                if (par_CAN[i].activ) trans_char(par_CAN[i].adr_can);
            trans_char(EOADR);
    };
    void  rec_valhex(void){//receptie  valori hexa  achizitionate  de
                           //  MINISPOT
            char i,j;
            trans_char(SOVAL);//initiere receptie
            for (i=0;i<NRCAN; i++)
                if(par_CAN[i].activ){
                        par_CAN[i].vallow=rec_char();
                        par_CAN[i].valhigh=rec_char();
                }
    }
    void main()

    {   int i;
            inportb(0x2F8);
            for (i=0;i<NRCAN;i++){
                par_CAN[i].vallow=0;
                par_CAN[i].valhigh=0;
                par_CAN[i].adr_can=i;par_CAN[i].activ=1;
            }
            par_CAN[5].activ=0;
            prog_8250();
            trans_adr();
            printf("continue:");
            getch();
            while (!kbhit()){
                rec_valhex();
                for(i=0;i<NRCAN;i++)
                   printf("canal %d %x  %x\n ",i,
                   par_CAN[i].vallow,par_CAN[i].valhigh);

            };
            trans_char('c');
```


Scanned with OKEN Scanner

```
}
```

Un program similar este implementat pe echipamentul SPOT avind urmatoarele functii:
1. O parte de initializare in care se receptioneaza numarul canalelor MUX pe care se face achizitia;
2. realizeaza achizitia pe canalele solicitate;
3. trimite valorile achizitionate pe interfata seriala S2 si reia programul de la pasul 2.

```
#define BOADR 0x31
#define EOADR 0x32
#define CANPORT1 0x21
#define CANPORT2 0x22
#define CANPORT3 0x23
#define CONVNOTRDY 0x40
#define SOVAL 0x33
struct {
      char vallow;
      char valhigh;
      char adr;
      char activ;
} par_can[24] ;

main()
{
        char cmd;
        int i;
        prog1();
        actmag();
/* initializari*/
        for(i=0;i(24;i++)
        {
                par_can[i].vallow=0x01;
                par_can[i].valhigh=0x04;
                par_can[i].aor=i;
                par_can[i].activ=0;
        }
/* se asteapta primirea areselor canalelor active*/
        while( rec_c() != BOADR);
        rec_a();
/*se comunica operatorului ce s-a receptionat*/
        for(i=0;i(24;i++)
                printf("%d",par_can[i].activ);
/*se repeta pasii  si 3 din algoritm pina se primeste de la
MASTER un caracter    SOVAL*/
        while ( rec_c() == SOVAL){
                achiz();
                for (i=0;i(24;i++) printf("canal %d %x %x\n",
                   i,par_can[i].vallow,par_can[i].valhigh);
                tr_val();
        };

}
```

Nu se prezinta in acesata lucrare din considerente de spat...

parte din functiile apelate cum sint: progi() (initializarea legaturii seriale), rec_a() (receptie adresa canal analogic impusa de PC), rec_c() (receptia unui caracter pe legatura seriala), tr_val() (transmisia valorii colectate de MINISPOT la PC). Pentru exemplificarea achizitiei de date la nivelul echipamentului MINISPOT sint date mai jos functiile C asociate.

```
achiz(){  /*achizitie canale active*/
int     i;
   for(i=0;i(24;i++){
        if (par_can[i].activ) {
                out(CANPORT1,par_can[i].adr);
                while(in(CANPORT1)&CONVNOTRDY);
                par_can[i].vallow=in(CANPORT3);
                par_can[i].valhigh=in(CANPORT2);
        }
   }
}

actmag(){/*activare magistrala proces*/
        out(0xc0,0x01);
        out(0xf1,0xfd);
}
```

In continuare se exemplifica modul de constructie a consolei operatorului de proces la nivelul PC utilizind concepte ale programarii orientate obiect. Elementele de interfata utilizate in proiectarea consolei operatorului de proces sint cele utilizate in consolele traditionale: butoane, comutatoare, inregistratoare (osciloscoape), afisari digitale, stari contacte (bec aprins/bec stins).

Realizind o ierarhizare a elementelor de interfata obtinem structura din figura 1. La baza ierarhiei se afla clasa aparat inzestrata cu functiile din dreapta (m- desemneaza o metoda). Aceste functii sint necesare pentru proiectarea interactiva a consolei. Pe urmatorul nivel al ierarhiei se afla doua clase: clasa **aparate de afisare** si clasa **aparate de comanda**. Aceste doua clase mostenesc elementele clasei de baza dar poseda si alte elemente ce le individualizeaza. Pe nivelul 3 al ierarhiei este rafinat nivelul 2.

Aparatele trebuiesc construite astfel incit sa permita o proiectare a consolei. De aceea ele sint dotate cu metode pentru pozitionarea lor pe ecran si pentru stabilirea dimensiunii lor. Prescurtarile au urmatoarea semnificatie: ls - limita superioara, li - limita inferioara, vr - valoarea reala de afisat, rata - periada de esantionare, buffer - zona de memorie pentru stocarea datelor esantionate.

Lucrind cu obiecte de genul celor prezentate proiectarea programului se apropie foarte mult de proiectarea schemelor electrice ceea ce este un avantaj deosebit . In plus obiectele construite devin instrumente software versatile .

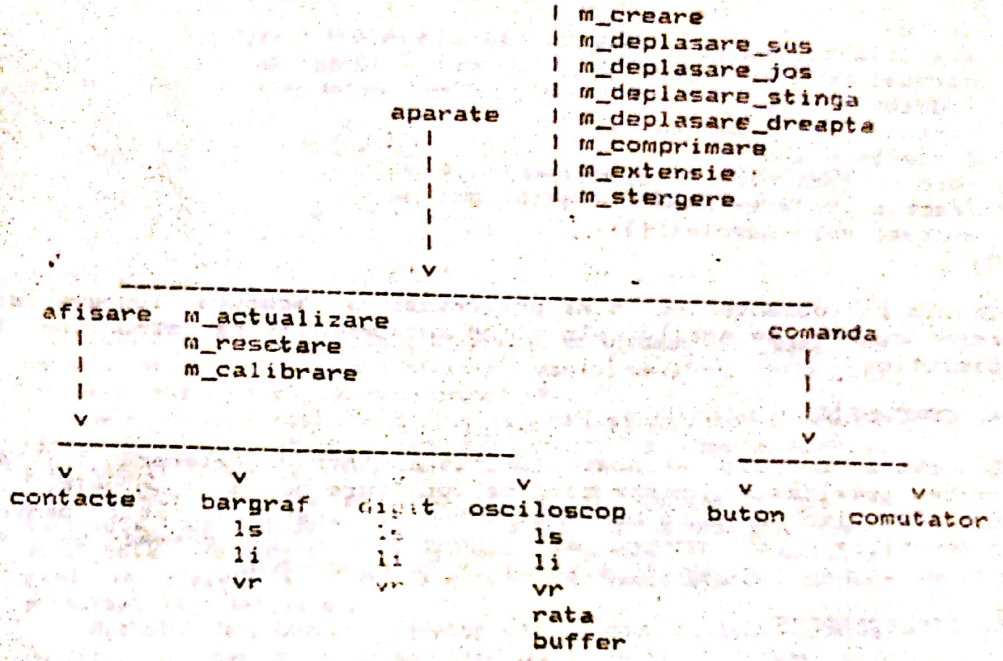


Fig. 1

Pentru implementarea in C++ s-a considerat ca la baza ierarhiei se afla clasa **aparat** cu urmatoarele elemente:

-date de tip **protected** (protejate) care pot fi folosite de metodele clasei dar si de metodele claselor derivate;

-metode definite cu cheia **virtual**. Pentru fiecare clasa derivata, metodele sint implementate diferit, punindu-se in evidenta proprietatea de polimorfism.

Fisierul care contine clasa aparat cu numele AP.H este prezentat mai jos.

```
/*****************************************************
clasa aparat - sta la baza ierarhiei de obiecte ce
                        se creaza pentru consola;
*****************************************************/

class aparat
{       protected:

        float   minvalue,maxvalue,sca,offset; //valoare  maxima,
                                              //minima,scara,
                                              //offset

        char *title;    /*texte*/
        float value;    /*valoare*/
        char *DisVSt;
        int LVMax;
```

```
        int poly[8];     //coordonatele aparatului
        int width,height; //latimea si inaltimea unui caracter

public:
 //initializarea aparatului
 virtual void Init(...) {} ;
 //desenare aparat
 virtual void Show(void){};
 //stergere aparat
 void virtual Hide(void){};
 //actualizare valoare
 virtual void Sample(){};
};
```

Celelalte obiecte nu sint prezentate in aceasta lucrare din considerente de spatiu, ele fiind implementate pe structura in discutie.

4. CONCLUZII

Structura propusa se poate constitui intr-un element initial pentru realizarea sistemelor de conducere pe doua nivele. In acelasi timp eate un bun instrument de instruire atit pentru studenti cit si pentru alti beneficiari ce doresc sa se perfectioneze in domeniul sistemelor de conducere.

5. BIBLIOGRAFIE

1. IBM, Personal Computer Hardware Reference
2. P. L. Anderrson, G. C. Anderson, Programming in ANSI C, 1989.
3. R. Lafore, Microsoft C Programming for the IBM, 1987.
4. Manual SPOT.
5. B. Eckel, Object - oriented programming with C++, 1990.

Scanned with OKEN Scanner

PACHET PENTRU ANALIZA PROGRAMELOR DE AUTOMAT PROGRAMABIL PE BIT (AP MICRO , AP 101 , AP 201)

as. ing. Călin Neaga (I.P.Iaşi)
as. ing. Corneliu Căileanu (I.P.Iaşi)

1) Introducere

Automatele programabile (AP) sînt un element de bază în domeniul automatizărilor industriale (maşini-unelte , instalaţii din industria chimică sau metalurgică etc.) .

Setul relativ restrîns de instrucţiuni şi caracterul relativ simplu al unor aplicaţii au dus uneori la desconsiderarea programării acestor echipamente .

Absenţa unor norme de lucru precise şi stricte a dus la existenţa (pe utilajele aflate în exploatare) a unor programe cu deficienţe care merg de la necorelarea cu ansamblul documentaţiei pînă la greşeli ce împiedică evident funcţionarea conform prescripţiilor iniţiale .

Pachetul de programe prezentat îşi propune analiza codului obiect ce ar comanda un automat programabil pe bit de fabricaţie românească (AP MICRO , AP 101-4K , AP 101-8K , AP 201) . S-a pornit de la premiza posibilităţii existenţei oricărei combinaţii de biţi , fie şi ca urmare a scrierii unui EPROM incomplet şters .

2) Descriere

2.1) Încărcarea codului obiect

Nu am dispus de un cititor de EPROM-uri aşa că am proiectat în acest scop un dispozitiv conectabil la interfaţa paralelă de pe placa de bază a unui microcalculator Cub-Z .

Codul obiect este citit cu un program scris în "C" şi în limbaj de asamblare şi depus într-un fişier pe disc flexibil .

Fişierele astfel create au fost transferate la un minicalculator de tip INDEPENDENT sau la un calculator tip IBM-PC , pe care s-au făcut toate prelucrările ulterioare .

2.2) Analizarea programelor de AP

Toate programele din pachet sînt scrise în limbajul "C" şi au ca rezultate finale fişiere text ce pot fi listate cu utilitarele sistemului de operare .

Principalii parametri ce trebuie indicaţi la lansarea programelor sînt tipul de AP (patru variante) , limitele zonei de iniţializare şi (eventual) opţiunea asupra părţii analizate (programul utilizator sau de iniţializare) .

Componentele principale ale pachetului de programe sînt :

(1) Dezasamblor

Programul prelucrează fişierul cu codul obiect de AP , rezultînd două fişiere listing : unul cu textul sursă şi celălalt

cu adresele la care s-a detectat un cod incorect (care nu poate fi dezasamblat) .

Textul sursă este prezentat în pagini corespunzînd la 50 de cuvinte de cod obiect , indicîndu-se adresa (în zecimal) , codul corespunzător (hexazecimal) , instrucţiunea (mnemonicul şi operandul) . La adresele cărora le corespund coduri incorecte apar în rubrica instrucţiunii semne de întrebare iar pentru adrese de salt în afara spaţiului real peresis se indică "JMP/JC ????" .

(2) Program de listare exclusiv a instrucţiunilor de salt

Se creează un fişier listing ce conţine numai instrucţiunile de salt din program , util în cazul folosirii intense a acestora sau ca bază de plecare în cazul necunoaşterii tipului de AP pentru un cod obiect dat (încercînd toate cele patru alternative s-ar putea trage concluzii utile) .

(3) Listare referinţe

Analizînd fişierul iniţial (codul obiect) , se creează un fişier cu variabilele folosite în program (intrări , ieşiri , memorii , temporizări - grupate în ordine crescătoare pentru fiecare tip) şi adresele la care apar (tot în ordine crescătoare pentru fiecare variabilă) .

Nu apar în listingul generat variabilele nefolosite în program .

(4) Detectarea ieşirilor comandate condiţionat

Programul creează un fişier listing în care sînt semnalate eventualele cazuri de ieşiri comandate prin instrucţiuni STO sau STOC în zone de program a căror parcurgere este condiţionată de instrucţiuni JC .

(5) Analiza instrucţiunilor de salt

Se creează un fişier conţinînd avertismente pentru următoarele situaţii : instrucţiuni de salt înapoi , salturi absurde (codul obiect corespunzător ar corespunde unui salt "pe loc") , sau eronate (pe al doilea cuvînt al unei instrucţiuni de salt - la AP 101-8K sau AP 201) .

(6) Analiza modului de utilizare a variabilelor

Din acest program rezultă un fişier intermediar în care se marchează modul în care sînt folosite (sau nu) variabilele I , M , T , E .

Se indică în comandă dacă se prelucrează programul utilizator sau cel de iniţializare , precizîndu-se limitele zonei de iniţializare .

(7) Semnalarea situaţiilor de folosire dubioasă a variabilelor M , T , E

Pornind de la fişierul creat în programul (6) , rezultă un listing în care se semnalează următoarele situaţii de utilizare potenţial greşită a unei variabile :

- apare atît într-o instrucţiune STO cît şi într-una STOC ;
- apare atît într-o instrucţiune STO/STOC cît şi într-o S/R ;
- apare în mai multe instrucţiuni STO sau în mai multe STOC ;
- apare într-o instrucţiune S dar nu şi într-una R (sau

invers) ;
- apare într-o instrucţiune STO/STOC dar nu şi în altă instrucţiune (diferită de S/R) ;
- apare într-o altă instrucţiune dar nu şi într-una STO/STOC/S (cu excepţia variabilei T032 la AP MICRO) .

Nu se face automat nici o legătură între programul utilizator şi cel de iniţializare deci trebuie consultate şi celelalte listinguri pentru eliminarea dubiilor .

(8) Crearea diagramelor de utilizare a variabilelor

Se porneşte tot de la fişierul intermediar creat în (6) , obţinîndu-se în final un listing cu tabelele de gestionare a variabilelor I , M , T , E .

Diagramele sînt de forma unor tabele de cîte 10 linii şi 10 coloane pe pagină .

Pentru intrări se marchează dacă sînt folosite în program sau nu . Pentru memorii , temporizări şi ieşiri se marchează modul de utilizare separat în instrucţiuni de tipul STO/STOC , S , R sau altele (blancuri pentru variabilele nefolosite) .

Diagramele se încheie cu ultima variabilă (din fiecare categorie) folosită , în limitele logice specifice fiecărui tip de AP .

Evident , programele (7) şi (8) se aplică separat pentru zona utilizator şi cea de iniţializare ale programului de AP .

9) Determinarea ecuaţiilor sursă

Din analizarea codului obiect rezultă ecuaţiile sursă corespunzătoare (cele care comandă efectiv maşina , nu neapărat cele dorite de proiectant sau de către producător) .

Prin comandă se indică limitele programului de iniţializare , care nu este luat în considerare (eventual poate fi tratat separat) .

În urma prelucrării rezultă un listing cu ecuaţiile sursă pentru toate instrucţiunile STO şi STOC (indicîndu-se adresa instrucţiunii respective) şi adresele pentru instrucţiunile S şi respectiv R , aceasta pentru toate variabilele astfel comandate (M , T , E) , în ordine crescătoare (M000 , ... , M511 , T000 , ... , T511 , E000 , ... , E511) .

O variantă a programului adaugă în listing şi ecuaţiile corespunzătoare instrucţiunilor S şi R .

Se admit mai multe instrucţiuni de acelaşi tip (STO/STOC/S/R) pentru aceeaşi variabilă .

Nu se iau în considerare eventuale ecuaţii înlănţuite prin instrucţiuni de salt . Dacă o ecuaţie nu începe cu o instrucţiune LD sau LDC (caz tipic : mai multe instrucţiuni STO succesive) , linia text va avea la început "!!!!" .

Pentru facilitarea urmăririi ecuaţiilor (şi ţinînd cont că operatorii nu au nici un fel de prioritate la această gamă de automate programabile) se folosesc în scriere paranteze , puse după cum urmează (nici o legătură cu parantezele din sintaxa limbajului de la AP 117) :
- între o instrucţiune OR/ORC sau AND/ANDC şi o instrucţiune XOR ;
- între o instrucţiune OR/ORC sau XOR şi o instrucţiune AND/ANDC .

3) Concluzii

Acest pachet de programe oferă o bază exactă şi consistentă

Scanned with OKEN Scanner

pentru exploatarea unui automat programabil pe bit , atît în faza de recepţie a unei maşini sau instalaţii cît şi la întreţinerea ulterioară , sau pentru detectarea şi eliminarea cauzelor ce duc la o funcţionare incorectă şi chiar la avarii grave .

Prin aplicarea întregului set unui program de AP se obţine o documentaţie aproape completă (excepţie făcînd , evident , sursa program) şi exactă , eliminîndu-se neconcordanţele cauzate de erorile de scriere sau de neactualizarea documentaţiilor .

De asemenea , forma de prezentare este foarte comod de urmărit şi se pretează la o multiplicare mai accesibilă decît cea uzuală (la heliograf) .

În acest mod se reduce substanţial munca obositoare şi migăloasă pe care ar presupune-o efectuarea operaţiunilor enunţate "manual" , specialistul din branşă trecînd direct la analiza şi corectarea deficienţelor .

La testări , timpul total necesar pentru aplicarea întregului lanţ de programe a fost de circa 8-9 minute pentru un kilocuvînt de cod obiect .

Costul unei astfel de prelucrări ar fi deci de cîteva zecimi de procent din costul total al unui utilaj echipat cu un AP , net inferior pierderilor produse prin avarierea sau chiar numai staţionarea acestuia .

Bibliografie

1) ### "AP 101. Manual de utilizare." , Automatica
2) ### "AP MICRO. Manual de utilizare." , Automatica

Scanned with OKEN Scanner

ANALIZA PROGRAMELOR PENTRU AUTOMATELE PROGRAMABILE PE CUVINT (AP 117)

as.ing. Călin Moaga (I.P.Iaşi)
as.ing. Corneliu Căilenau (I.P.Iaşi)

1) Introducere

Automatele programabile (AP) şi-au cîştigat un rol binemeritat în domeniul automatizărilor industriale datorită avantajelor oferite (flexibilitate în proiectare , robusteţe în exploatare etc.) .

După automatele programabile pe bit au apărut cele programabile pe cuvînt (la care se adaugă pe lîngă setul de instrucţiuni secvenţiale şi operarea cu instrucţiuni numerice) , reprezentînd un nivel de comandă superior .

Lipsa unor mijloace de proiectare asistată de calculator şi a unor norme stricte a condus în unele cazuri la programe cu deficienţe mergînd de la neconcordanţe în documentaţie pînă la greşeli pe cît de obscure , pe atît de contraproductive .

Echipamentul de programare uzual (consola de programare CPP 286) , deşi are un asamblor mult mai riguros pentru AP pe cuvînt decît pentru cele pe bit , oferă mijloace de control foarte limitate , inerente unui soft înscris în EPROM-uri .

Programele prezentate în continuare îşi propun analiza codului obiect ce ar comanda un AP pe cuvînt de fabricaţie românească (AP 117) . Am păstrat principiile generale pe care ne-am bazat şi în analiza programelor de AP pe bit .

2) Descriere

Pachetul de programe este scris în întregime în limbajul C şi poate fi rulat pe un minicalculator de tip INDEPENDENT sau pe un compatibil IBM-PC . Rezultatele prelucrărilor sînt fişiere text ce pot fi listate cu utilitarele sistemului de operare .

Componentele principale sînt :

(1) Dezasamblor

Programul dezasamblează codul obiect pentru AP 117 şi creează două fişiere listing , unul cu programul sursă în paşini corespunzînd la cîte 50 de cuvinte de memorie şi celălalt cu adresele la care s-a detectat un cod incorect (nu poate fi dezasamblat corect sau este vorba de o greşeală de logică în folosirea operanzilor) .

O variantă a programului primeşte ca parametri în linia de comandă şi adresele între care se face dezasamblarea .

(2) Program de listare exclusiv a instrucţiunilor de salt

Se analizează codul obiect pentru AP şi se creează un fişier listing în care apar instrucţiunile de salt găsite în program ("JMP" , "JMPI" , "CAL" , "END" , "RPT") .

(3) Analizarea instrucţiunilor de salt

Se creează un fişier în care se semnalează salturile dubioase (pe o instrucţiune "+" , "x" etc.) sau incorecte (pe unul din cuvintele unei instrucţiuni numerice altul decît primul) .

De asemenea , se verifică adresele , indexul şi numărul de repetiţii pentru instrucţiunile "RPT" .

(4) Urmărire folosire variabile

Programul creează un fişier binar intermediar în care se marchează modul de utilizare a variabilelor secvenţiale şi numerice .

Operaţiunea se face separat pentru programul de iniţializare şi pentru programul utilizator .

Se marchează şi variabilele care nu apar explicit în programul sursă dar vor fi utilizate în execuţie prin includerea lor în bucle "RPT" . Dacă prin indexări repetate s-ar depăşi spaţiul logic de adresare , se emite un mesaj de avertisment la terminal .

(5) Crearea diagramelor de utilizare a variabilelor

Folosind rezultatul de la (4) , se obţine un fişier listing cu diagrame în pagini de cîte 10 linii x 10 coloane pentru fiecare tip de variabilă în parte . Fiecare diagramă se încheie cu ultima variabilă efectiv folosită în program .

Pentru variabilele secvenţiale se marchează următoarele variante de utilizare :

- într-o instrucţiune "=" sau "=/" ;
- într-o instrucţiune "S" ;
- într-o instrucţiune "R" ;
- în altă instrucţiune secvenţială ;
- ca variabilă de lansare ;
- ca variabilă de marcaj ;
- pentru Carry sau Borrow ;
- ca sursă sau destinaţie la o deplasare ;
- într-un registru pas-cu-pas ;
- folosită prin intermediul unei instrucţiuni "RPT" .

Pentru variabilele numerice se indică următoarele posibilităţi de utilizare :

- ca sursă ;
- ca destinaţie "MN/EN" sau numai "EN" ;
- sînt alterate într-o instrucţiune ;
- ca sursă pentru o instrucţiune "JMPI" ;
- ca destinaţie prin "ATV" ;
- folosită prin intermediul unei instrucţiuni "RPT" .

La ieşirile numerice se semnalează utilizarea unui canal special (care nu-şi modifică valoarea) sau a unuia fictiv .

Diagramele prezintă rezultatele separat pentru programul utilizator şi pentru programul de iniţializare .

(6) Semnalarea cazurilor dubioase de gestionare a variabilelor

Folosind rezultatele de la (4) se obţine un listing în care se prezintă următoarele situaţii de eroare evidentă sau de dubiu :

- o intrare ar fi folosită ca destinaţie ;
- o memorie , temporizare sau o ieşire secvenţială :
  - apare în mai multe instrucţiuni "=" sau "=/" ;
  - apare în "S" dar nu şi în "R" ;
  - apare în "R" dar nu şi în "S" ;

- apare ca destinaţie şi nu este folosită ca sursă ;
- apare ca sursă şi nu este folosită ca destinaţie ;

- o ieşire numerică :
- apare ca sursă ;
- apare ca destinaţie în altă instrucţiune decît "TRAD" , "TRDD" , "TRDS" ;
- se foloseşte un canal special sau unul fictiv .

Nu se face nici o conexiune între programul utilizator şi cel de iniţializare deci trebuie corelate rezultatele cu celelalte listinguri .

(7) Listare referinţe

Pornind de la codul obiect pentru AP 117 , se creează un fişier cu referinţele variabilelor (atît secvenţiale cît şi numerice) în ordinea lor crescătoare .

Nu apar în listing variabilele nefolosite în program ; nu se separă iniţializarea de programul utilizator . În cazul folosirii într-o instrucţiune numerică , se indică adresa apariţiei efective , nu cea de început a instrucţiunii .

(8) Determinarea ecuaţiilor (secvenţiale) sursă

Prin acest program se creează un fişier cu ecuaţiile secvenţiale corespunzătoare instrucţiunilor "=" , "=/" , "S" şi "R" , indicîndu-se şi adresele lor . Se admit mai multe ecuaţii de acelaşi tip pentru aceeaşi destinaţie ; nu se separă programul utilizator de iniţializare .

Destinaţiile apar în listing în ordinea crescătoare (M000 , ... , M511 , T000 , ... , T511 , E000 , ... , E511) a celor comandate în program .

O variantă a programului indică pentru instrucţiunile "S" şi "R" numai adresele la care apar .

Dacă o ecuaţie nu începe cu "!" , "!/" sau cu deschiderea unei paranteze (deşi asamblorul consolei de programare CPP 286 încearcă să excludă această posibilitate) , linia text va începe cu "!!!!" .

În scrierea ecuaţiilor ,parantezele sînt cele din sintaxa limbajului , deci trebuie să se ţină cont de prioritatea operatorului "*" faţa de operatorul "+" . Dacă este cazul , la începutul ecuaţiei se scriu numărul de paranteze deschise cerut de aritmetică şi nu una singură , cum apare în programul sursă .

3) Concluzii

Creşterea complexităţii echipamentelor utilizate nu s-a corelat la noi cu o îmbunătăţire substanţială a metodelor de lucru . Modul de lucru "manual" a dus în multe cazuri la scăpări ce dau multă bătaie de cap chiar numai pentru detectarea lor , remedierile fiind extrem de dificile .

O simplă greşeală de scriere în documentaţie impune un efort serios chiar pentru o simplă comparaţie cu conţinutul memoriilor EPROM cu care se lucrează , şi aceasta cu condiţia de a avea la îndemînă o consolă de programare , lucru care nu se întîmplă întotdeauna .

Prin realizarea acestui pachet de programe ne-am propus să oferim o documentaţie exactă şi consistentă pentru un program dat de AP pe cuvînt . Se elimină în acest fel munca migăloasă şi de uzură pe care ar presupune-o abordarea fără sprijinul calculatorului a detectării şi analizării cauzelor ce conduc la o funcţionare defectuoasă a unui utilaj echipat cu AP .

De asemenea , această documentaţie ar reprezenta un sprijin substanţial în activitatea curentă de întreţinere .

Rezultatele acestor prelucrări permit să se treacă direct la analiza şi corectarea deficienţelor .

Nu am avut la dispoziţie pentru testare un program mai amplu dar putem estima timpul de calcul necesar aplicării întregului lanţ de prelucrări la aproximativ 7-9 minute pe kilocuvînt de cod obiect . Evident , costul unei astfel de prelucrări ar fi nesemnificativ în raport cu costul total al unui utilaj sau chiar numai cu cheltuielile de întreţinere .

Bibliografie

1) ### "AP 117. Manual de utilizare." , Automatica
2) ### "CPP 286. Manual de utilizare." , Automatica

Scanned with OKEN Scanner

MODELAREA SI SIMULAREA CIRCUITELOR DIGITALE

Gorgan Dorian, Dobricean Sorin,
Rusu Dan, Ieremie Dana
Institutul Politehnic Cluj-Napoca

1. Introducere

Simularea si testarea schemelor logice corespunzatoare placilor echipate cu circuite digitale necesita construirea unui model. Construirea modelului se realizeaza interactiv prin tehnica meniurilor sau se poate utiliza un meniu realizat anterior pe baza informatiilor de structura extrase dintr-un fisier realizat cu mediul ORCAD. Modelul cuprinde informatii despre topologia placii, informatii legate de conectarea circuitelor, atribute ale circuitelor dintr-o biblioteca, valorile acestor atribute sau parametri de simulare si redactare a rezultatelor simularii.

Se prezinta o varianta de simulare orientata pe evenimente. Un eveniment se defineste ca fiind o modificare de stare logica a unui semnal intr-un nod de circuit. Gestiunea in timp a evenimentelor se face cu ajutorul unui secventiator. Secventiatorul determina evenimentul curent si creaza o lista cu evenimentele urmatoare determinate de evenimentul curent. Simulatorul pastreaza o lista de evenimente posibile impreuna cu momentele producerii lor. El alege timpul de producere minim si executa actiunea declansata. Ca urmare lista de evenimente posibile se modifica in urma adaugarii de noi evenimente dupa care se revine la selectarea unui nou eveniment. Experimentul de simulare se termina la epuizarea timpului de simulare programat.

2. Modelarea sistemelor digitale

Topologia interna reprezinta imaginea plachetei externe in memoria calculatorului. Sint reprezentate entitati ca si: cai de legatura intre pinii diferitelor componente de pe placheta, componentele fizice utilizate pe placheta impreuna cu legaturile dintre pini. Aceasta reprezinta structura fizica a plachetei. Topologia mai cuprinde o multime de atribute cu valorile lor, asociate pinilor: semnatura logica etalon, semnatura logica amprenta corespunzatoare diferitelor defecte utilizate in testare, valoare logica, nivel de tensiune, timp de propagare, fan-out maxim, etc.

In conceperea topologiei interne au stat la baza urmatoarele criterii:

- reprezentarea cit mai fidela a plachetei cu evitarea redondantei si inconsistentei datelor;
- posibilitatea accesului cit mai rapid la elementele din model, indiferent de sistemul de operare si limbajul de programare utilizat;

- posibilitatea modularizarii topologiei si existenta unor operatii de partitionare, reuniune, salvare si incarcare in memorie a diferitelor module.

Topologia interna trebuie sa asigure informatiile necesare parcurgerii plachetei sau schemei logice dupa orice algoritmi pe acelasi nivel ierarhic - toti pinii unei componente, toate caile din topologie, toti pinii unei componente tip sau fizice, toate atributele unui pin, etc; cit si intre nivele ierarhice diferite - pin <--> componenta tip <--> componenta fizica,cale <--> pin <--> atribute, etc.

Topologia interna contine urmatoarele entitati: cai (noduri de circuit), componente fizice de circuit (U1, Y23 utilizarea concreta a unui integrat digital), componente tip (CD8400, 74LS08, 8086, 4116), pinii componentelor tip (CLK, VCC, D, T, Q), pinii componentelor fizice (1, 14, 6), atributele si valorile acestora pentru pinii fizici si pinii tip (IN, OUT, TS, time, delay).

De exemplu, caile topologiei sint descrise printr-o lista inlantuita de structuri, fiecare corespunzind unei cai.Fiecare structura contine informatii de parcurgere a topologiei, informatii necesare simularii cit si testarii: nume cale, pointer la inceputul listei ce constituie vectorul secventiatorului pentru respectiva cale, pointeri la expresiile Poage utilizate in determinarea vectorilor de test, starea de defect a nodului, numarul pinilor din cale, starea de eveniment a nodului la momentul curent de evaluare, pointer la lista pinilor din cale, pointer la urmatoarea cale din lista.

3. Construirea topologiei

Toplogia se creeaza in fisiere text folosind programul NETLIST din produsul ORCAD cu optiunea RACALREDAC si cu optiunea /p pentru a se preciza pinii circuitelor prin numar si nu prin nume.Astfel se obtin doua fisiere, unul cu extensia .CMP continind asociatiile nume fizic componenta - nume tip componenta; si unul cu extensia .NET continind conexiunile din topologie. Pentru generarea automata a programelor de test este necesar sa fie precizate caile de intrare si iesire din topologie (liniile la conector) si tipul circuitelor (analog sau digital) din topologie.

4.Secventiatorul

Secventiatorul contine vectorii de stare asociati fiecarui nod de circuit.

Notam cu vi[0] starea curenta a semnalului din nodul i la momentul curent t si cu vi[1] starea urmatoare a semnalului din nod la momentul t+T, unde T este cuanta de timp de evaluare. Atunci vi[dt] va fi starea la momentul t+dt≤T. vi[dt] se stabileste la momentul t evaluind functia de transfer corespunzatoare pinului de iesire care dicteaza valoarea logica a semnalului din nodul i.Pentru pinii de intrare care apar ca argumente pentru functia logica, valorile lor se considera ca valori v[0] din secventiator, fiind valorile semnalelor din

nodurile de intrare din care fac parte acesti pini.
Algoritmul de evaluare a semnalului dintr-un nod de circuit si de determinare a unor evenimente viitoare este urmatorul:

```
evaluare(nod)
    daca eveniment(nod)
        pentru fiecare pin de intrare din nod
            $determina functia logica de transfer
            $evalueaza functia de transfer, folosind
             valorile curente din secventiator pentru
             toti pinii de intrare care apar ca argumente
             ale functiei de transfer
            $determina intirzierea corespunzatoare
             pinului de intrare curent
            $determina pinul de iesire care preia
             valoarea functiei de transfer
            $determina nodul din care face parte pinul de
             iesire
            $introduce in secventiator pe pozitia
             corespunzatoare intirzierii valoarea
             functiei de transfer
      altfel
          $nimic
```

Determinarea functiei de transfer se face relativ simplu:
-se determina componenta fizica din care face parte pinul de intrare considerat;
-se determina componenta tip corespunzatoare componentei fizice;
-se determina numele functiei logice asociate componentei tip (NOT, AND3, OR2).

5. Simulatorul

Este un instrument foarte util in studiul functionarii unei scheme logice digitale realizate cu circuite combinationale sau secventiale de tip MSI. Este un simulator orientat pe eveniment, are trei stari logice: 0 (zero logic), 1 (unu logic), X (stare indiferenta), si starea de eveniment corespunzatoare celor doua tranzitii: Up (tranzitie low --> high), Down (tranzitie high --> low). Permite simularea a doua tipuri de defecte:

- nod de circuit blocat in 0 logic;
- nod de circuit blocat in 1 logic.

Simulatorul poate fi utilizat in doua moduri:
a. Apelat direct. In acest mod se poate utiliza intregul set de functii ale simulatorului, cum ar fi:
- editarea interactiva a schemelor logice;
- incarcarea, salvarea unei topologii editate;
- editarea in diverse moduri a secventelor binare aplicate la intrarea circuitelor simulate;
- afisarea topologiei, bibliotecii de circuite, valorilor logice a semnalelor din nodurile de circuit la fiecare pas de

Scanned with OKEN Scanner

simulare. Se pot edita in varianta actuala 40 de semnale de intrare pe durata a 256 pasi de simulare;

- programarea optiunilor de simulare, editare, afisare sau generare automata a vectorilor de test;
- simularea functionarii circuitului cu diverse optiuni programate;
- punerea in evidenta a comportarii circuitului la diferite frecvente de lucru, a aparitiei hazardului;
- generarea automata a vectorilor de test pentru defecte si puncte de vizualizare a defectelor programabile.

b. Apelat din preprocesor. In acest mod se utilizeaza facilitatea secventiatorului de a genera automat vectori de test pentru un circuit dat. Se incearca astfel gruparea automata a cit mai multe circuite intr-un acelasi modul de test.Rezulta o marire a vitezei de testare functionala si o reducere a numarului de puncte de test necesare intregii placi.

Lansarea directa a Simulatorului dintr-un meniu principal se face prin selectia comenzii Simulate.Dupa lansarea Simulatorului apare pe ecran meniul principal sub forma de bara orizontala.Cursorul luminos va fi pozitionat initial pe comanda Data File.Cu tastele sageti se selecteaza una din comenzi.Meniul bara contine urmatoarele comenzi: Data File, Topologie, Simulation, Options, Test si Quit.Pentru fiecare comanda selectata se afiseaza pe ecran un meniu specific.Comenzile principale au urmatoarele organizari ierarhice ale subcomenzilor:

Data File

- **1.EditData** - permite editarea datelor de intrare in simulator;
  - **1.1.Up** - afisarea liniilor 1-20 din matricea de datelor de intrare;
  - **1.2.Down** - afiseaza liniile 21-40 din matricea datelor de intrare;
  - **1.3.Left** - afiseaza pagina precedenta paginii curente din matricea de date;
  - **1.4.Right** - afiseaza pagina urmatoare paginii curente din matricea de date;
  - **1.5.Jump** - salt la o pagina specificata;
  - **1.6.Set** - programarea unei linii de date de intrare cu o valoare logica 0 sau 1 specificata;
  - **1.7.Random** - programarea cu valori aleatoare a unei linii de date pe un intervai specificat;
  - **1.8.Edit** - se trece intr-un regim de editare a intregii matrici de date de intrare cu 0 sau 1.Deplasarea cursorului se face cu tastele sageti;
  - **1.9.Quit** - se revine in meniul principal al Simulatorului;
- **2.SaveData** - salveaza intr-un fisier cu numele specificat continutul matricei de date de intrare;
- **3.RetrieveData** - incarca dintr-un fisier specificat matricea cu datele de intrare;

Scanned with OKEN Scanner

Topologie

1.EditTopo – permite editarea interactiva de la consola a topologiei placii echipate;
- 1.1.Place – insereaza o componenta fizica in topologie Se specifica tipul si numele componenetei;
- 1.2.Connect– conecteaza doi pini sau un pin cu o linie' de date de intrare;
- 1.3.Unconect – desface legatura unui pin;
- 1.4.Delete – contine o clasa de comenzi de stergere selectabile dintr-un submeniu;
  - a.Component – sterge o componenta fizica specificata prin nume;
  - b.All – sterge toata shema logica;
- 1.5.Show – afisari.
  - a.Component – afiseaza o componenta precizata prin nume.Afiseaza atribute si conectarile pinilor;
  - b.Way – afiseaza o cale (nod de circuit) specificata prin nume.Afiseaza toti pinii din cale si atributele caii;
  - c.AllWays – afiseaza toate caile din topologie. Dupa afisarea unei cai se continua cu celelalte prin apasarea unei taste. Daca se tasteaza <ESC> afisarea se abandoneaza;
  - d.Library – afiseaza numele componentelor tip din topologie;

2.OrcadTopo – incarca o topologie dintr-un fisier de pe disc;

3.SaveTopo – salveaza o topologie din memorie pe disc;

4.SelectTopo – selecteaza topologia curenta cu care se lucreaza din multimea de topologii incarcate;

5.Partition – partitioneaza topologia al carei nume este cerut in module de test;

Module

1. Show – afisari referitoare la modul
   - a. Component – afiseaza lista de componente existente intr-un modul precizind atit numele fizic cit si tipul acestora;
   - b. Ways – afiseaza caile de intrare si caile de iesire dintr-un modul;
   - c. Pins – afiseaza caile si pinii circuitelor pe care au fost pozitionate cuie de test;

Scanned with OKEN Scanner

Simulation

1.Assign – asigneaza unui nod din schema logica o pozitie in afisarea formelor de unda din nodurile de circuit la fiecare pas de simulare;

2.ShowAssign – afiseaza asignarea curenta;

3.ClearAssign – sterge asignarea curenta;

4.Options – permite programarea unor optiuni referitoare la afisarea rezultatelor unui experiment de simulare;

4.1.FirstCycle – programarea primului pas de simulare care se afiseaza.Poate fi in domeniul 1..256 acelasi cu adincimea matricii de date de intrare.Pe durata unui pas de simulare (Cycle) vectorul binar de la intrarea circuitului ramine neschimbat;

4.2.LastCycle – programeaza ultimul pas de simulare care se afiseaza;

4.3.PagePause – permite programarea pauzei intre ecranele ce afiseaza pasii de simulare;

4.4.WidthSubcycle – programeaza latimea in pixeli la afisarea unui subciclu.Un subciclu se defineste ca fiind rata de evaluare a evenimentelor din nodurile de circuit;

4.5.Subcycle/Cycle – programeaza latimea unui pas de simulare (Cycle) in numar de cuante de timp de evaluare;

4.6.MarkCycle – marcheaza la afisare pasii de simulare;

4.7.MarkSubcycle – marcheaza la afisare fiecare cuanta de timp de evaluare noduri de circuit;

Test – genereaza automat vectorii de test pentru un modul a carui topologie si circuite se cunosc;

1.SetFault – se precizeaza numele componentei fizice, numele pinului si valoarea defectului;

2.ClearFault – anuleaza asignarea de la SetFault;

3.ShowFault – afiseaza numele caii si tipul defectului;

4.SetVisiblePin – se precizeaza numele componentei si pinul pentru care se asigura observarea defectului;

5.GenTestArray – afiseaza vectorii de test;

Quit – revenirea in meniul principal.

6. Fisierele de sabloane si de diagnoza

In urma generarii vectorilor de test rezulta fisierele de sabloane si de diagnoza. Fisierul de sabloane este un fisier binar, obtinut automat cu ajutorul Preprocesorului. Acest fisier are extensia implicita .WPL si contine urmatoarele informatii :

- numele formei de unda - nume simbolic asociat sablonului;
- numele componentei careia ii este asociat sablonul;
- numarul de semnale pe care se aplica sablonul;
- numele acestor semnale;
- numarul de secvente care se aplica circuitului;
- frecventa cu care se aplica secventele;
- secventele propriu-zise;

Fisierul de diagnoza este un fisier binar in care se specifica pasul din secventa de test si defectele care pot aparea in acel pas cu secventele binare care le pun in evidenta. Acest fisier are extensia implicita .DIA. Este creat automat de catre Preprocesor sau interactiv cu ajutorul editorului de sabloane.

7.Preprocesorul

Pentru modulele de test se face un test functional. Partitionarea placii in module de test se face automat de catre Preprocesor.

Pentru fiecare modul de test se determina:

- descrierea topologica cuprinzind circuitele digitale integrate ce compun modulul si lista de conexiuni dintre pinii componentelor fizice;
- sabloanele de test si defectele puse in evidenta de catre fiecare sablon. Se creaza un fisier cu sablonul de test si un fisier de diagnoza. Fisierul de diagnoza precizeaza defectele care se pun in evidenta cu fiecare vector de test;
- lista nodurilor de circuit in care se vor amplasa cuie de test (driver sau senzor) din patul de cuie.

Partitionarea placii se face pentru circuitele a caror functionare se modeleaza in biblioteca de functii logice. In continuare se incearca extinderea unui modul cuprinzind si alte circuite de pe placa, dar in asa fel incit la bornele modulului sa se poata propaga orice defect singular din nodurile interne ale modulului.

Pentru determinarea vectorilor de test necesari pentru a pune in evidenta defectele din modul la pinii de iesire ai modulului, se apeleaza Generatorul Vectorilor de Test din cadrul Simulatorului.

Vectorii de test, pentru fiecare modul, se memoreaza in fisiere cu sabloane. Pentru fiecare modul extins se creaza de asemenea, cite un fisier de diagnoza. In fisierul de diagnoza se inscriu raspunsurile de pe placa sub test corespunzatoare fiecarui defect din interiorul modulului.

Scanned with OKEN Scanner

8.Concluzii

Simulatorul de circuite poate fi utilizat cu succes in proiectarea asistata a schemelor digitale atit discrete cit si integrate. Varianta actuala a simulatorului urmeaza sa fie imbunatatita dupa verificarea mecanismelor de lucru cu semnale cu patru stari logice (0, 1, X si Z) si puteri diferite, doua tranzitii (Up si Down), sesizarea starilor de hazard si a tranzitiilor simultane, a timpilor diferiti de propagare, a conflictelor in noduri, a defectelor simultane, etc.

9.Bibliografie

1. Vladutiu, M., Crisan M. - Tehnica testarii echipamentelor automate de prelucrare a datelor, Editura FACLA, Timisoara 1989.
2. Breuer, M. A., Friedman, A. D. - Diagnostigs and Reliable Design of Digital Systems, Computer Science Press Inc.
3. Chang, H. J., Manning, E., Metze, G. - Fault Diagnosis of Digital Systems, Willey-Interscience, New-York, 1970.

Scanned with OKEN Scanner

ASAMBLOR PENTRU SISTEME CU TRANSPUTER T414/T800

Autori: Barsan Corneliu
Institutul Politehnic Iasi
Albulet Lucretia
Bejenariu Silviu
Roman Mariana
Barsan Laura
Institutul de Informatica Teoretica, Iasi

INTRODUCERE

Lucrarea prezinta un asamblor pentru transputerele T800 si T414 realizat la Institutul de Informatica Teoretica Iasi. Exista in prezent mai multe sisteme cu transputere T414/T800 dezvoltate ca extensii pentru calculatoare IBM-PC sau ca sisteme independente. Aceste sisteme se livreaza impreuna cu medii de programare complexe incluzind compilatoare, asambloare, etc.

Asamblorul prezentat in lucrarea de fata face parte din categoria instrumentelor de dezvoltare si este destinat in principal proiectantilor de sisteme cu transputere.

In prezent, scrierea unui asamblor reprezinta o sarcina de complexitate medie, fara un caracter de noutate deosebit. Totusi, anumite particularitati ale setului de instructiuni ale transputerelor fac ca problema generarii optimale de cod sa nu fie chiar triviala.

Solutiile gasite pentru optimizarea codului sint, deasemenea, prezentate in lucrare.

1. TRANSPUTERELE T414/T800 - PREZENTARE GENERALA

Un transputer este un microprocesor RISC proiectat special pentru medii multiprocesor. Un chip (transputer) contine: un procesor, 2kb memorie RAM (4kb pentru T800), patru canale seriale DMA, doua timere si procesor de virgula flotanta (numai T800). Un executiv (cablat hard) asigura executarea proceselor specificate de doua liste de procese corespunzatoare a doua nivele de prioritate.

Setul de registre este format in principal din: trei registre generale utilizator organizate ca o stiva -A,B,C-, un registru operand O, un registru pointer de instructiuni I, un registru pointer spatiu de lucru W, doua perechi de registre pentru gestiunea listelor de procese.

Instructiunile sint de lungime fixa pe 8 biti: 4 biti pentru codul operatiei si 4 biti pentru operand. Setul instructiunilor

include 13 instructiuni de baza, doua instructiuni de prefixare si o instructiune speciala opr de indirectare.

Pregatirea in registrul operand a operanzilor care nu pot fi reprezentati numai pe patru biti se face prin intermediul a doua instructiuni pfix, nfix (pfix deplaseaza la stinga patru pozitii registrul O) dupa care operandul pe patru biti din instructiune este plasat in cimpul 0-4 din O.

O instructiune de forma operatie #expresie va fi tradusa de asamblor intr-un numar corespunzator de instructiuni pfix sau nfix urmate de o instructiune operatie.

Exemplu: Instructiunea j 100 se traduce in:

```
21   pfix 1
20   pfix 0
01   j    0
```

Completarea setului de instructiuni se asigura prin intermediul instructiunii operate precedata de o instructiune de prefixare. Instructiunea operate determina interpretarea continutului O ca instructiune.

2. LIMBAJUL DE ASAMBLARE RECUNOSCUT, SINTAXA DE DESCRIERE

Asamblorul recunoaste limbajul specificat de firma INMOS [1] complementat cu o serie de directive de asamblare. Sintaxa liniei de program este urmatoarea:

[eticheta] delim operator delim [operand][delim! comentariu]

2.1 Cimpul eticheta

Acest cimp contine un sir de caractere alfanumerice impreuna cu caracterul "-" din care primul caracter fiind obligatoriu o litera. Lungimea maxima a unei etichete este de maximum 40 de caractere . Literele mici se disting de cele mari.

O eticheta introduce un nume la care se asociaza o valoare. Aceasta reprezinta, de regula, valoarea contorului de locatii la linia respectiva. Exceptie de la aceasta regula constituie liniile cu operatorul .eq, caz in care valoarea e data de cimpul operand.

2.2 Cimpul operator

In cimpul operator pot apare :
- -mnemonice instructiuni
- -nume directive
- -nume macrouri

In acest cimp nu se face distinctie intre literele mari si mici.

Scanned with OKEN Scanner

2.2.1 Mnemonice instructiuni

Sint acceptate mnemonicele din manualul IMMOS.

2.2.2 Numele directivelor

2.2.2.1. Directiva .align

Sintaxa : eticheta .align

Determina alinierea codului la o adresa para . Cind contorul de locatii contine o valoare impara se insereaza in cod un octet nul.

2.2.2.2. Directivele .db , .dw , .ds

```
Sintaxa :    [eticheta]    .db  expresie
             [eticheta]    .db  "sir de caractere"
             [eticheta]    .dw  expresie
             [eticheta]    .ds  expresie
```

Directiva db determina rezervarea unui octet de memorie, octet care se initializeaza cu valoarea data de cimpul operand.
Daca in cimpul operand apare o constanta de forma "sir de caractere" se genereaza un vector continind caracterele sirului.

Directiva .dw determina rezervarea unui cuvint (32 de biti) continind valoarea data de cimpul operand.

Directiva .ds se foloseste pentru rezervarea de zone de memorie.Lungimea in octeti a zonei rezervate e data de valoarea cimpului operand.
Cimpurile operand pentru directivele .dw, .db pot contine expresii evaluabile la prima sau a doua trecere , iar cimpul operand al directivei .ds trebuie obligatoriu sa contina o expresie evaluabila la prima trecere .

2.2.2.3. Directiva .eq

Sintaxa : nume .eq expresie

Permite definirea unui simbol si atribuirea unei valori simbolului respectiv. Numelui definit in cimpul eticheta i se asociaza valoarea expresiei din cimpul operand, expresie care trebuie sa fie evaluabila la prima trecere.
Simbolul definit cu directiva .eq poate fi folosit prin numele sau in expresiile din cimpurile operand.

Scanned with OKEN Scanner

2.2.2.4. Directivele .list si .nolist

Sintaxa : .list
.nolist

Determina activarea/dezactivarea generarii fisierului listing.

2.2.2.5 Directiva .org

Sintaxa: .org expresie

Directiva .org schimba valoarea contorului de locatii cu valoarea data de expresia operand, expresie care trebuie sa fie evaluabila la prima trecere.
Directiva .org nu poate fi folosita la rezervarea spatiului de memorie.

2.2.2.6. Directiva .include

Sintaxa : .include nume-fisier

Directiva .include are ca efect continuarea citirii de catre asamblor la intrare a liniilor din fisierul specificat in "nume-fisier".
Fisierul care se include poate contine la rindul sau directive .include , permitindu-se maximum 6 nivele de adincime.

2.2.2.7. Directivele .if .else .endif

Sintaxa :
```
.if expresie
...
corp instructiuni 1
...
.else
...
corp instructiuni 2
...
.endif
```
sau
```
.if expresie
...
corp instructiuni 1
...
.endif
```

Directivele .if , .else , .endif implementeaza facilitatea de asamblare conditionata .
Daca expresia e de valoare nenula se asambleaza liniile din "corp instructiuni 1". Daca expresia are valoare nula se asambleaza liniile din "corp instructiuni 2".

Scanned with OKEN Scanner

2.2.2.8. Directivele .macro .endm .label

Sintaxa :
```
.macro    nume-macro    arg1,arg2,...argk
.label    et1,et2,..etk
...
corp macro
...
.endm
```

Directiva .macro specifica numele unui macrou si parametrii formali ai macro-ului. Liniile din "corp macro" sint cele cuprinse intre directivele .macro si .endm.

Directiva .label declara etichetele locale din interiorul unui macrou pentru care asamblorul va genera la fiecare invocare a macro-ului etichete distincte in vederea evitarii definitiilor multiple.

2.2.2.9. Directivele .t414 .t800

Sintaxa :
```
.t414
.t800
```

Au rolul de a specifica transputerul pentru care se face asamblarea codului.

Initial se considera implicit T414.

2.2.3. Nume macro-uri

Aparitia in cimpul operator a unui nume care nu e mnemonica sau nume directiva, determina cautarea numelui respectiv in lista de macro-definitii . Apelul unui macrou are ca efect inserarea liniilor corpului macro cu substituirea simpla a parametrilor formali cu cei actuali . De asemenea , se genereaza etichetele locale daca exista o directiva .label .

2.3. Cimpul operand

Contine o expresie aritmetica formata din constante , nume simbolice si operatori. Se folosesc urmatorii operatori in ordinea precedentei :

```
( )      - paranteze stinga , dreapta
~  -     -complement fata de 2 si minus unar
>> <<    -deplasare dreapta , stinga
&  |     - si  sau (pe bit)
*  /     -produs  , impartire
+  -     -adunare , scadere
```

Ca operanzi se pot folosi nume simbolice definite cu directiva .eq sau ca etichete si constante numerice.

Constantele se specifica astfel :

- constante zecimale dddd Ex: 1234
- constante hexazecimale $hhhh Ex: $12A6

- constante octale @0000 Ex: @177
- constante binare %bbbb Ex: %1001
- constante ASCII "aaaa" Ex: "sir"

2.4. Cimpul "comentarii"

Sintaxa : !"comentariu"

Este optional si e ignorat de asamblor . Se introduce prin caracterul "!" in prima coloana.

3. STRUCTURA DE PRINCIPIU A ASAMBLORULUI OPTIMIZAREA CODULUI

Asamblorul lucreaza in 4 faze principale: preprocesare, trecerea 1, optimizare, trecerea 2.

Preprocesarea este destinata tratarii directivelor de asamblare conditionata si macroprocesarii. Trecerea 1, in principal, asigura crearea tabelei de simboli, iar in trecerea a doua se genereaza codul.

Necesitatea unei faze de optimizare este ilustrata de urmatoarele exemple:

Exemplul 1:

Instructiunea ldc $A12 se poate traduce in oricare din urmatoarele variante echivalente:

```
2A   pfix $A          20   pfix $0
21   pfix $1          20   pfix $0
42   pfix $2          20   pfix $0
                      20   pfix $0
                      20   pfix $0
                      2A   pfix $A
                      21   pfix $1
                      42   ldc  $2
```

Prima varianta nu incarca toti cei 32 de biti ai registrului operand. A doua varianta, care completeaza cu 0 la stinga registrul operand, are avantajul ca indiferent de operand, lungimea codului este constanta, dar nu este optimala.

Dependenta lungimii codului generat de valoarea operandului determina o serie de dificultati.

Exemplul 2:

```
1          et1:
2               cj   et2
3          et3: j    et1
           ...
n          et2:
```

Secventa anterioara e dificil de tradus din urmatoarele motive: la prima trecere, in tratarea liniei 2, valoarea simbolului (adresei) "et2" nu este cunoscuta si nu se poate

determina lungimea codului generat pentru instructiunea "j et2", decit daca se considera offsetul pe 32 de biti.

Contorul de locatii nu se poate incrementa [illegible] simbolul "et3" nu se poate introduce in tabela cu valoarea corespunzatoare. De asemenea, nu se poate calcula offsetul et3-et1 necesar in generarea codului pentru instructiunea "j et1".

O solutie imediata pentru completarea tabelei de simboli cu valori contor-program (p_c) corecte este sugerata in [1] si consta in mai multe treceri asupra programului. La prima trecere se realizeaza doar o estimare a valorilor p_c ale simbolilor din tabela. Urmatoarele treceri nu genereaza cod, ci recalculeaza aceste valori obtinind estimari mai bune pina la obtinerea valorilor p_c corecte. Solutia este inacceptabila sub raportul timpului de asamblare.

In asamblorul descris de prezenta lucrare am implementat o metoda mai eleganta care evita trecerile multiple si, in plus, permite expresii aritmetice oarecare in cimpul operand.

Principial, metoda consta in urmatoarele:

1. La evaluarea unei expresii aritmetice care apare ca operand se recalculeaza si un atribut expresie care precizeaza daca valoarea ei este absoluta (formata numai din constante) sau depinde de una sau mai multe valori de tip adresa (eticheta).

2. Pentru expresiile care nu sint absolute se introduc in tabela de simboli pseudoetichete, iar o forma a expresiei se pastreaza in vederea recalcularilor ulterioare.

3. La prima trecere se considera ca valorile tuturor expresiilor care nu pot fi calculate sint reprezentabile pe 4 biti. Se obtine astfel o prima estimare a valorilor din tabela de simboli.

4. Urmeaza o faza de optimizare in care se parcurge tabela de simboli de mai multe ori pina la stabilizarea valorilor p_c.

In acest scop tabela de simboli este organizata ca o structura dublu inlantuita. Tabela poate fi exploatata atit ca o structura hash (la interogare), cit si ca o lista inlantuita simpla de simboli a caror ordine in lista este ordinea aparitiei lor in program. Pe parcursul unei treceri prin tabela de simboli (lista) se identifica prin intermediul pseudoetichetei prima expresie care nu este absoluta si se reevaluaza. Diferenta pozitiva sau negativa fata de valoarea anterioara se aduna la toti simbolii de tip eticheta care urmeaza in program.

Procesul continua cu urmatoarea expresie care nu este absoluta, pina la parcurgerea completa a tabelei de simboli. Se obtine astfel o estimare mai buna.

Se reia cu o noua parcurgere pina nu mai apar diferente.

Ultimele asignari ale valorilor p_c la simboli sint folosite in trecerea a doua care genereaza codul.

Scanned with OKEN Scanner

Concluzii

Lucrarea are un acracter informativ si nu prezinta toate detaliile de proiectare si implementare. Acestea pot fi gasite in [2] sau pot fi cerute autorilor. Algoritmul de optimizare este original si are caracter de noutate.

Eventualele neclaritati sau omisiuni nu sint voluntare si se datoreaza spatiului limitat de prezentare.

Bibliografie

[1] INMOS-Transputer instruction set - a compiler writer's guide
[2] I.I.T. Fil. Academiei, Iasi - MAT Macroasamblor pentru transputer T414, T800 - Specificatii de definitie

Scanned with OKEN Scanner

SISTEME EXPERT IN COMUNICATIE CU BAZE DE DATE

Autor: dr. ing. D. Carstoiu
Institutul Politehnic Bucuresti

1. INTRODUCERE

Lucrarea prezinta o modalitate prin care pot fi crescute capabilitatile unui sistem de baze de date (DBMS) prin utilizarea unui sistem expert (SE). Sint cunoscute in principal trei moduri de integrare a sistemelor expert si bazelor de date obtinind ceea ce se numeste **baze de date inteligente, sisteme expert cu capacitate crescuta si sisteme expert/baze de date comunicative.** Principala diferenta intre cele trei modele este marcata de modul de cuplare si de alocare a controlului in sistemele ce le integreaza. Principala problema a bazelor de date inteligente este cea a accesului la inregistrari dupa diferite criterii, alegerea celui mai bun criteriu de acces putind fi formulata ca o problema de decizie a celei mai bune selectii dintr-un subset al alternativelor posibile. Tehnicile de utilitate teoretica cum sint conflictele de rezolutie, metodele interactive si cele cu intindere mare pot fi inlocuite cu succes in multe din problemele complicate specifice bazelor de date. Conceptele 'fuzzy' utilizate inevitabil atit in sistemele expert cit si in rationarea umana pentru luarea deciziilor se pot utiliza cu succes in problematica bazelor de date.

Un sistem de baze de date impreuna cu un sistem expert bazat pe logica 'fuzzy' poate manipula concepte vagi cerute de ambele aplicatii pe linga functiile normale alocate acestora. Aplicatiile specifice acestor categorii de probleme se vor construi intr-un mediu constituit dintr-un, sistem expert bazat pe logica 'fuzzy' impreuna cu un modul de interfatare la un program specific de baze de date.

2. INTERACTIUNEA SISTEM EXPERT BAZE DE DATE

O baza de date este o colectie de date intr-o forma organizata ce poate furniza date catre sistemul expert prin inntermediul sistemului de manipulare a datelor. Exista mai multe posibilitati de interactiune intre sistemul expert si baza de date, interactiuni ce pot fi grupate in trei categorii in conjunctie cu gradul de cuplare si alocare al controlului. Se disting astfel sistemele expert cu capacitate marita, bazele de date inteligente si sisteme expert/baze de date in comunicatie. Desigur ca aceste trei categorii pot fi impartite in viitor in alte clase si nivele asa cum se arata in fig.1.

In **sistemele expert cu capacitate marita** facilitatile extinse de manipulare a datelor sint implementate la nivelul sistemului expert. In acest caz datele sint furnizate sistemului expert de baza de date atunci cind sint cerute, fie sint transmise sistemului expert inaintea oricarei activitati a acestuia ca la transferul imaginii intr-o fotografie la minut. Este modul in

Scanned with OKEN Scanner

care baza de date este pusa la dispozitia sistemului expert in totalitate.

Prin inglobarea componentelor deductive intr-un sistem de baze de date se obtin **bazele de date deductive si inteligente.** Componenta deductiva poate creste eficienta si functionalitatea sistemului de baze de date prin optimizarea cerintelor, pastrarea integritatii si a altor facilitati. Formal o baza de date deductiva este definita ca un triplet

D = (Th, IC, DO) in care,

* Th reprezinta o teorie cu un set finit de axiome logice, si un set finit de metareguli ce descriu modul de operare cu axiomele teoriei particulare.
* IC reprezinta un set finit de restrictii de integritate ce trebuiesc sa fie satisfacute de baza de date, constind dintr-un set finit de formule ale calculului cu predicate.
* DO reprezinta setul finit al constantelor peste care teoria este definita.

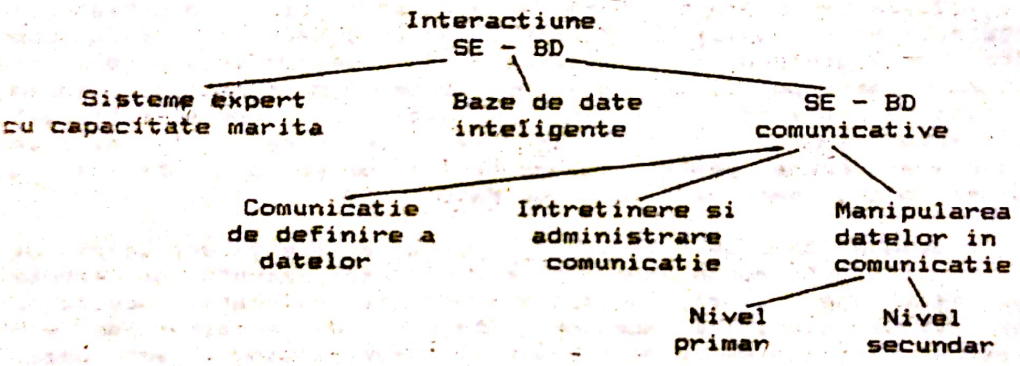


Fig. 1.

Reuniunea (Th U IC) dintre teoria (Th) si restrictiile de integritate (IC) trebuie sa satisfaca conditia de consistenta. Formulele atomice si regulile de deductie sint doua dintre axiomele tipice ale unei teorii. Formulele atomice intre n-upluri si relatii formeaza baza de date extensionala pe cind regulile deductive, reprezentate prin declaratii in forma logica determina o baza de date intensionala.
Ca exemplu se considera o baza de date formata din doua relatii extensionale: **arie_locatie (id_arie, judet)** si **trafic_conditii(judet, ruta)**, pentru care se poate defini o relatie intensionala ce arata conditiile de trafic pentru aria in discutie astfel:

$$\forall\, id\_arie\ \forall\, judet\ \forall\, ruta$$
$$arie\_locatie(id\_arie, judet)\ \&\ (trafic\_conditii(judet, ruta) \longrightarrow arie\_trafic\_conditii(id\_arie, ruta)$$

A treia categorie **sisteme expert/baze de date comunicative** are

avantajul asigurarii independentei celor doua componente, fiind posibile citeva forme de comunicatie intre acestea. Utilizarea unei baze de date existente face ca gradul de complexitate al comunicatiei intre SE si DB sa fie mult mai mare. La rindul sau comunicatia poate fi clasificata in trei clase:

* **comunicatia de definire a datelor**, prin care SE poate altera structuri si componente ale BD. Daca cunostintele privind dezvoltarea BD cum sint cele despre normalizare sint continute de sistemul expert atunci o BD poate fi construita sau modificata automat.

* **intretinerea dazei de date si administrarea comunicatiei** este situatia in care SE poate rezolva sarcini administrative cum sint analiza stocarii, gestionare backup, reorganizarea si intretinerea datelor.

* **manipularea datelor in comunicatie**, prin care SE trimite comenzi pentru adaugare, inserare, stergere catre baza de date. Utilizarea numai a acestui tip de schimb de informatie face ca un SE sa aiba forta unei baze de date deductive. O instructiune logica a unei relatii intensionale intr-o baza de date deductiva este de fapt o regula ce poate fi impartita in doua parti: prima parte este o definire a unui n-uplu scris in termenii calculului relational ce este echivalent cu partea IF a regulii, a doua parte fiind data de numele si atributele n-uplului pentru relatia intensionala ce este echivalenta cu partea THEN. Este motivul pentru care instructiunile logice pentru relatiile intensionale intr-o baza de date deductiva pot fi inlocuite prin reguli intr-un SE, impreuna cu un limbaj de cereri relationale suportat de DBMS.

Pentru a treia clasa de comunicare sint necesare doua nivele ale manipularii datelor. Primul nivel este specific bazelor de date, fara sa tina cont de operatiile in SE. Cu acest nivel SE manipuleaza si selecteaza inregistrari prin el insusi facindu-l echivalent cu un SE de mare capacitate. Al doilea nivel presupune utilizarea operatorilor de inalt nivel si a altor limbaje suportate de DBMS. La acest nivel pentru accesul la date SE transmite cereri catre DBMS utilizind partea IF a regulii ca o conditie de selectie. In acest mod SE nu va selecta sau sorta date prin el insusi.

Lucrarea de fata este destinata celei de a treia clase de cuplare care asigura avantajul utilizarii unei BD existente, deoarece nu este practic sa se construiasca un nou sistem de BD in care sa se transfere datele inainte de a fi utilizat. Astfel DBMS impreuna cu setul de date este vazut ca un sistem independent. Un modul special care asigura interfata intre SE si DBMS este necesar, modul care accepta cereri fuzzy de la SE sau poate fi utilizat independent daca utilizatorul furnizeaza cereri in limbajul acceptat.

3. ARHITECTURA SISTEMULUI

Un SE 'fuzzy' pentru aplicatii cu BD (SEBD) este constituit in principal din trei componente si anume:

* Sistemul Expert Fuzzy (SEF)

Scanned with OKEN Scanner

* Modulul de Comunicare cu Baza de Date (MCBD)
* Baza de Date (BD)

Structura SEBD este prezentata in fig.2. SEF are o structura tipica SE bazat pe reguli. Este compus dintr-un modul de achizitia cunostintelor, un mecanism de inferenta si o baza de cunostinte fuzzy. Cunostintele in domeniu sint extrase de la expertii umani si transferate sistemului prin intermediul modulului de achizitie a cunostintelor. Regulile si faptele pot fi stocate in baza de cunostinte prin utilizarea si a termenilor fuzzy. Mecanismul de inferenta este raspunzator de utilizarea cunostintelor si deducerea concluziilor bazate pe cunoasterea exacta sau inexacta.

MCBD asigura interfata intre SEF si BD si face ca SEF sa poata utiliza BD. Sistemul poate manipula date fuzzy ce pot fi direct stocate in baza de date si sint utilizate la rationarea fuzzy in timpul consultarii. Faptul ca SEF manipuleaza cereri lingvistice fuzzy este foarte important in luarea deciziilor pentru multe probleme reale.

4. REPREZENTAREA CUNOSTINTELOR SI INFERENTE

In SEF asa cum s-a prezentat mai sus cunostintele sint reprezentate prin reguli de forma:

IF propozitie-antecedent
THEN propozitie-succedent
WITH CERTAINTY factor-certitudine

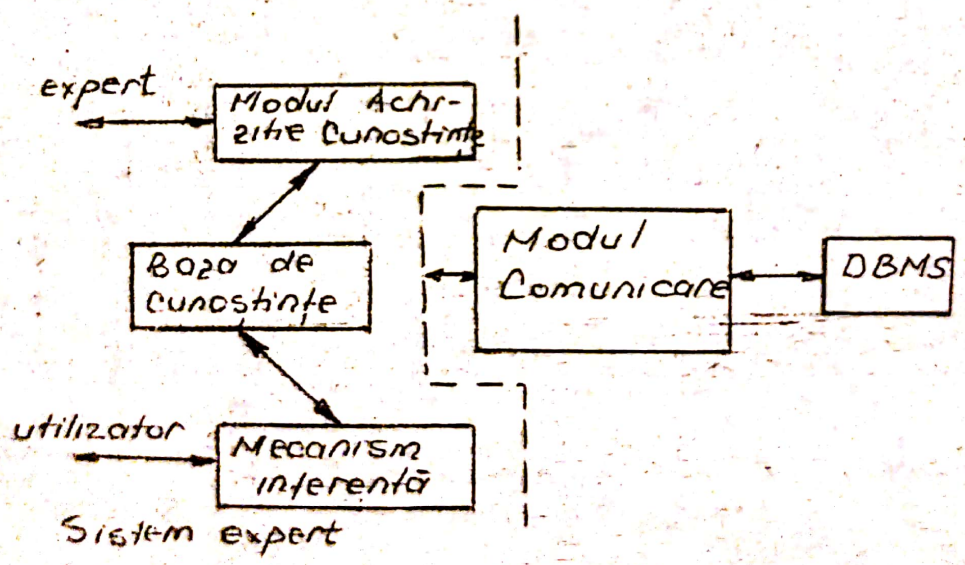


Fig.2

Astfel fiecare propozitie este reprezentata ca "obiect" cu

"valoare", ca de exemplu;

```
IF        hirtia indicatoare este rosie
THEN      solutia este acida
WITH CERTAINTY  1.0
```

Astfel un obiect ce este o unitate de baza in baza de cunostinte (BC) poseda un numar de proprietati ce sint transmise in faza de achizitie a cunostintelor. Se dau mai jos citeva exemple ale acestor proprietati:

Proprietate	Semnificatie
Tip	Tipul obiectului
Flag-cerere	Valoarea este obtinuta prin intrebari adresate utilizatorului sau prin deducerea din alte reguli si fapte
Flag- fuzzy	Indica faptul ca obiectul are o valoare fuzzy sau nu

Asa cum in rationarea umana se utilizeaza concepte fuzzy, SEF permite termeni fuzzy in reguli si este capabil sa realizeze rationamente aproximative, ca de exemplu;

```
Regula    IF        masina este puternica
          THEN      dezvolta viteza mare
Fapte     Masina este foarte puternica
-------------------------------------------
Concluzie  Viteza dezvoltata este foarte mare
```

In acest exemplu puternic, foarte puternic si viteza sint termeni fuzzy. Fiecarui termen fuzzy ii corespunde un set fuzzy, setul fuzzy reprezentind concluziile este obtinut prin operatiile realizate intre setul fuzzy reprezentind faptele si relatiile fuzzy reprezentate in regula. Propozitii multiple si termeni mixti fuzzy si non-fuzzy sint permisi in partea antecedent si succedent a regulii.

O metoda de rationare este cea de inlantuire inainte, asa ca atunci cind valoarea unui obiect este ceruta SEF incearca mai intii sa o caute in baza de date. Daca valoarea este lipsa atunci sistemul invita utilizatorul sa o introduca direct determinind astfel o legatura dinamica intre componentele sistemului. Datele sint cautate in baza de date numai atunci cind este necesar un proces de inferenta la nivelul SE. Ca exemplu se poate considera situatia starii sanatatii si a sansei de insanatosire a unui bolnav. Daca starea de sanatate este apreciata dupa greutate corelata cu inaltime se poate aprecia sansa insanatosirii ca fiind foarte mica, mica, mare, foarte mare. Desigur ca este important de observat ca temenii fuzzy sint recomandabil a fi utilizati cind este necesara utilizarea unor descriptori numerici cu mai multe valori.

O caracteristica a acestui sistem este abilitatea de acces la informatiile relevante din baza de date la sfirsitul consultarii

acesteia. Aceasta este obtinuta prin generarea unui set de cereri fuzzy catre baza de date dupa ce sistemul expert trage concluzia. In timpul procesului de inferenta, restrictiile ce constituie cereri la baza de date sint culese. Astfel ca, in regula

```
IF                      persoana este adult
THEN SET_RESTR          virsta)21 GREUTATEA 0.8 AND
                        poate vota
```

SET_RESTR este o actiune ce indica faptul ca urmatoarea propozitie este o restrictie. Daca regula este evaluata cu succes fraza virsta)21 este extrasa si devine una din restrictii. GREUTETEA este un cuvint cheie ce specifica faptul ca factorul greutate este 0.8 in acest caz, el fiind in general reprezentat ca un set fuzzy cu valori reale cuprinse intre 0 si 1. Din inlantuirea tuturor regulilor se obtine cererea pentru baza de date. Cererile fuzzy returneaza o gama larga de informatii a caror apartenenta este determinata de continutul bazei de date. Pentru formularea:

```
(sportiv_rec(nume) () (and(grad"bun" 1.0)))
```

adica lista numelor sportivilor cu rezultate bune, pentru care a fost definita distributia fuzzy dupa punctaj astfel

grad()	0	10	20	30	40	50	60	70	80	90	100
set fuzzy	0	0	0	0	0	0.1	0.2	0.4	0.7	0.9	1.0

se obtine o lista a sportivilor ordonata dupa valorea variabilei fuzzy asociate. Se observa ca valorile gradelor de participare sint determinate de distributia participarii datelor stocate in baza de date.

Daca sint aplicabile mai mult de o conditie fuzzy apartenenta la o clasa este obtinuta prin combinarea conditiilor rezultate in mai multe moduri:

1. Calcul minimum/maximum (and/or)
2. Combinarea evidentei ca in teoria confirmarii (comb)
3. Interogare si capturare marime (poll)

Daca se presupune ca a_1, a_2, ..., a_n sint gradele de apartenenta, se obtine dupa aplicarea operatorilor definiti mai sus.

AND $f(a_1, a_2, \ldots, a_n) = \min(a_1, a_2, \ldots, a_n)$

OR $f(a_1, a_2, \ldots, a_n) = \max(a_1, a_2, \ldots, a_n)$

COMB $f(a_1, a_2) = a_1 + a_2(1-a_1)$

$$f(a_1, a_2, \ldots, a_n) = \begin{cases} a_1 & n=1 \\ a_1 + a_2(1-a_1) & n=2 \\ f_n(f_{n-1}(\ldots), a_n) & \end{cases}$$

POLL $f(a_1, a_2, \ldots, a_n) = (a_1, a_2, \ldots, a_n)/n$

Orice combinatie a celor patru operatori este permisa a fi folosita intr-o expresie fuzzy. Inregistrarile cu gradul de apartenenta final zero sint eliminate din setul de iesire. Logica fuzzy este superioara metodelor de selectie multicriteriale prin flexibilitatea manipularii relatiilor dintre aceste criterii. De asemenea, criteriile fuzzy sint subiective si trebuiesc formulate cu multa grija.

Modulul de comunicare cu baza de date preia cererile formulate de SEF pentru obtinerea informatiilor din baza de date, activitatile sale putind fi etapizate in trei faze:

* translatarea cererilor SEF in limbajul specific BD.
* preluarea datelor de la BD.
* procesarea datelor cerute pentru incadrarea in clase de apartenenta, ordonarea acestora in ordine descrescatoare a gradului de apartenenta si eliminarea celor cu grad de apartenenta zero.

Sintaxa cererilor fuzzy utilizate de SE este urmatoarea:

```
QUERY           ::=((nume_relatie) LIST_SIMB, COND_FUZZY,
                                NON_FUZZY_CON)
LIST_SIMB       ::=ST_EXP I ((num_cimpi),..,(num_cimpi))
COND_FUZZY      ::=FUZZY_CON I nil
ST_EXP          ::=(ST_OP NON_FUZZY_CON)
NON_FUZZY_CON   ::=(OPERATOR TERM1....TERMj)
FUZZY_CON       ::=(FUZZY_OP FUZZY_EX1....FUZZY_EX2)
ST_OP           ::=max(num_cimp) I
                   min(num_cimp) I
                   total(num_cimp) I
                   average(num_cimp) I
                   count
TERM            ::=NON_FUZZY_CON
                   (COMPARATOR (num_cimp) (valoare)) I
                   (COMPARATOR (num_cimp) ST_EXP)
FUZZY_EX        ::=FUZZY_CON I FUZZY_TERM
FUZZY_TERM      ::=((num_cimp) (membru al distributiei)
OPERATOR        ::=and I or
FUZZY_OP        ::=and I or I comb I poll
COMPARATOR      ::=(<, (=,),>=)
```

Sint necesare citeva precizari asupra caracteristicilor DBMS pentru a se asigura integrarea in acest sistem. S-a presupus ca DBMS nu este specializat dar accesul la informatii se realizeaza prin combinarea comenzilor START_STREAM, FETCH, GET, sau echivalente. Inregistrarile ce satisfac criteriul de selectie sint incluse in sirul inregistrarilor specificate cind se executa START_STREAM, un pointer de sir este plasat inaintea primei inregistrari a sirului. Avansul in sir se realizeaza prin comanda FETCH, iar valoarea si numele cimpului poate fi obtinut prin GET. Procesul se repeta pina cind toate inregistrarile din sir sint explorate.

5. CONCLUZII

In acest mod datele pentru inferenta sint obtinute de la o baza de date intr-o forma organizata permitind prelucrarea cu sistemul expert a unei mari cantitati de date. Acest mod de abordare, cit si utilizarea rationamentelor fuzzy poate oferi suportul de rezolvare a multor probleme ale lumii ,reale. O aplicatie ce devine interesanta este cea a diagnosticului genetic unde se dispune de o baza de date organizata dar care ocupa 14 Mo, cantitate de informatie ce nu poate fi utilizata in alt mod.

6. BIBLIOGRAFIE

1. Teza de doctorat, D. Carstoiu 1989.
2. Rule-based Expert Systems, B. G. Buchanam, 1984.
3. Fuzzy Set Theory and itsAplications, H. J. Zimmermann, 1986.
4. Expert Systems and Fuzzy Systems, C. Negoita, 1985.
5. Data & Knowledge Engineering, 1989-1990, North-Holland.

Scanned with OKEN Scanner

SISTEME SUPORT DE DECIZIE CU APLICATII IN MEDICINA

Autori Dr. ing. Dorin Carstoiu
Institutul Politehnic Bucuresti

1. INTRODUCERE

Inteligenta artificiala este un domeniu relativ nou al stiintei calculatoarelor ce isi propune sa rezolve problemele intr-o maniera apropiata de modul de gindire uman. Dintre instrumentele de lucru ale inteligentei artificiale, sistemele expert ocupa un loc central mai ales prin varietatea aplicatiilor lor. Dezvoltate in ultimul deceniu, sistemele expert au revolutionat viziunea referitoare la tipurile de probleme ce pot fi rezolvate folosind calculatorul.

Organizate putin diferit fata de familiarele programe algoritmice, sistemele expert pot folosi tehnici euristice si rationamente "fuzzy" pentru a ajunge la rezultate. In plus astfel de sisteme incorporeaza un volum mare de cunostinte asociate unor domenii specifice. Sistemele expert sint programe foarte performante in domeniile lor inguste de expertiza pentru care au fost elaborate. Bine intretinute, relativ ieftine, cu mare grad de portabilitate si fiabilitate ridicata, devin din ce in ce mai eficiente cu trecerea timpului.

2. SISTEME SUPORT DE DECIZIE

O categorie mai larga privind utilizari frecvente ale tehnicii de calcul in domeniul medical este cea a sistemelor suport de decizie (SSD). In ciuda entuziasmului manifestat de multi utilizatori, in ultimii ani dezvoltarile de succes se refera la domenii restrinse si bine definite.

Ca regula generala, decizia umana este o operatie complexa ce nu presupune intotdeauna o intelegere profunda a procesului. Deliberarile sint adesea neclare si parti ale fenomenului nu pot fi conchise individual. Constructia SSD poate de cele mai multe ori sa ceara calculatoarelor facilitati de simulare a comportamentului uman. In viitor, informatiile cerute de calculatoare cer compresiunea colectiilor de date prin tehnici structurale ce sint incomode din punctul de vedere al utilizatorului. Astfel, dezvoltarea unui SSD valid necesita un proces lung cu reveniri si noi acumulari de date. Un exemplu graitor este proiectul INTERNIST ce a fost dezvoltat continuu mai multi ani. Ca si la alte produse, calitatile unui astfel de sistem trebuiesc analizate si validate in urma exploatarii intr-un timp indelungat. In cazul sistemelor cu aplicatii clinice validarea are un important caracter particular. Datorita faptului ca sistemele suport intentioneaza a fi un ultim pas in decizia medicala, multe informatii din diverse surse vor fi combinate in deciziile clinice de diagnosticare si terapie. Validarea unui astfel de task devine foarte dificila datorita complexitatii

deosebite a sistemului. Numai un sistem ce functioneaza intr-un context particularizat pentru un grup individualizat de utilizatori poate beneficia de opinii calificate asupra capacitatii sale.

Sigur ca pentru o analiza atenta a SSD cu aplicatii clinice trebuiesc examinate dupa destinatie: in diagnosticare, tratament, prognoza, etc.. De asemenea raspunsul la o serie de intrebari trebuie sa tina seama de toti factorii implicati in utilizarea sistemului, adica, pacient, informatician, medic , societate. Considerentele legate de utilizare presupun evaluarea metodelor fundamentale referitoare la algoritmii utilizati, cunostintele stocate si sursa acestora, cit si a caracteristicilor operationale. Exista la ora actuala programe de dezvoltare a mediilor sistem pentru limbaje orientate profesional, programe ce includ cercetari si colaborari multidisciplinare in sfera informaticii si medicinii.

In context clinic, obiectivele utilizarii SSD trebuiesc judecate independent de performantele sistemului, realizind pe cit posibil compromisul intre complexitatea problemelor medicale si simplitatea manipularii noilor mecanisme de decizie. Multe dintre sistemele utilizate in domeniul medical sint destinate furnizarii cunostintelor specializate, pornind de la premisa ca fie utilizatorul este inudat de o cantitate mare de informatii, fie ca problema duce la situatii nefamiliare acestuia. Din acest motiv se pot distinge patru tipuri de sisteme:

* sisteme vizind condensarea informatiei pentru facilitarea prelucrarilor ulterioare, avind avantajul largirii capacitatii personalului uman in rezolvarea de probleme complexe.
* sisteme capabile sa ofere utilizatorilor nespecializati suportul deciziilor, purtind din acest motiv si numele de sisteme consultative.
* sisteme de invatare utilizate pentru educarea si perfectionarea utilizatorilor, cu avantajul cresterii competentei profesionale a acestora.
* siteme de informare si reorganizare, cu avantajul acesului simplu la literatura specifica problemei.

Scopul final al aplicarii oricarei noi tehnologii este cresterea grijii fata de sanatate, in sensul luarii de catre medic a celei mai bune sau, a celei mai putin eronate decizii. Desigur ca, utilizarea unui astfel de sistem va determina doctorul sa cheltuiasca mai mult timp cu pacientul, influientind si modificari in structura serviciilor de sanatate, noi analize ale raportului cost/eficienta.

Cum esenta unor astfel de sisteme este tocmai stocarea cunostintelor, se remarca validitatea mult mai mare a cunostintelor fata de date. Este adevarat ca aceste cunostinte au diverse costuri. Se poate defini o cunostinta ca o abstractizare a experientei obtinute de om, ce este direct sau indirect derivata din date si informatii. Domeniile cunoasterii se pot imparti in doua si anume domenii inchise si domenii deschise. Un domeniu inchis este rar si foarte restrins si nu poate acoperi, nici

partial, un domeniu clinic . Este deci necesar ca utilizatorii sa cunoasca caracteristicile propriului domeniu, focalizarea si identificarea cerintelor aplicatiei sistemice, tehnicile probabilistice si multimile vagi.

3. EXTRAGEREA CUNOSTINTELOR

Cunostintele specifice domeniului de expertiza sint stocate in diverse moduri, modelele cel mai des intilnite fiind regulile de productie, retelele semantice precum si reprezentare orientata obiect. Deductia in forma sa cea mai simpla consta din doua parti: partea din stinga, premiza sau ipoteza si partea din dreapta, consecinta sau actiunea. Astfel, ipoteza implica concluzia sau premiza actiunea. Diferenta intre aceste doua categorii este minima in sensul ca satisfacerea premizei declanseaza automat actiunea.

De exemplu, un medic stie ca tensiunea normala pentru un barbat de 30 ani cu inaltimea de 1,80 m care cintareste 75 kg este de 11/7. Acest fapt se poate exprima ca o regula prin care se atribuie o valoare unei variabile TENSNORM in functie de valoarea atribuita variabilei VIRSTA.

```
IF:   VIRSTA ( 30
      AND  INALTIMEA = "1,80"
      AND  GREUTATEA = 75
THEN: TENSNORM = "11/7"
```

Procesorul de inferenta evalueaza premiza determinind valoarea de adevar a expresiei, si in functie de aceasta, declanseaza actiunea. Un sistem expert simplu folosit pentru determinarea greutatii si tensiunii normale a unei persoane pentru care se dau virsta, sexul si inaltimea contine relativ putine reguli in timp ce un sistem expert pentru aplicatii largi poate contine sute de reguli.

Cunostintele de diagnosticare sint esentialmente empirice si deriva din observatiile asupra diferitilor pacienti individuali. Deci prima evaluare a surselor de cunoastere este marcata de observarea bazei de cunostinte. Este necesar in acest moment sa se stabileasca criteriile de incluziune si excluziune a elemantelor de cunoastere, stiind ca baza de cunostinte este cea care afecteaza puternic validitatea SSD.

Datele despre pacienti sint colectate prospectiv si retrospectiv. De regula inregistrarile retrospective au o alta destinatie si sint incomplete. Rezultatele negative sint rareori inregistrate.

In cartile medicale, articole, alte publicatii si lucrari stiintifice se reprezinta multiple cunostinte despre modul de stabilire a diagnosticului. Trebuie remarcat faptul ca informatiile provenind din carti au ca punct de plecare simptomatologia fara a oferi explicatii pe traseul invers; diagnostic - simptom. Experienta specialistilor individuali sau de grup reflecta rationamentul pentru diagnosticarea in domeniul

Scanned with OKEN Scanner

specificat, operind dupa anumite scenarii, de multe ori diferite dar cu aceleasi rezultate. Datele sint usor organizate prin program, informatiile obtinute in acest mod putind suplimenta cunostintele achizitionate. Un aspect important se refera la tratarea cazurilor inchise si a celor extreme. Sint relevante aspectele legate de clasificarea pacientilor dupa virsta, sex, etnii, criterii geografice si socioeconomice.

Cele mai multe SSD au fost construite pe baza datelor colectate de un grup mic de cercetatori. Transferul de la un grup de utilizatori la altul, prin caracterul diferit al observatiilor poate duce la cresterea acuratetetii rezultatelor. Reproductibilitatea observatiilor va trebui evaluata ca parte a acizitiei de cunostinte si apreciata astfel pentru adaugarea de noi date pe timpul testarii de sistem. Gradul transferabilitatii bazei de cunostinte depinde de mai multi factori, valorile predictive ale SSD trebuind sa fie testate cind sint aplicate unui nou mediu.

Pentru validarea rezultatelor SSD este necesar sa se cunoasca citeva rezultate standard, general acceptate, cu care sa se compare iesirile sistem. Sint posibile doua situatii de acceptare atunci cind se obtine un raspuns corect pentru problema cu obiectivul cautat sau cind concluziile corespund in mare parte cu cele ale expertului uman (sau grupurilor de experti umani). Este de asemenea dificil de clasificat rezultatele SSD ca fiind corecte sau false, intrucit solutia unei probleme se obtine ca o combinatie de rationamente si propozitii cu diferite grade de certitudine. Concluzii alternative pot fi corecte la diferite nivele de incredere si o solutie corecta la un caz poate acompania un diagnostic incorect.

Este greu de statuat faptul ca opiniile personale sau practica colectiva serveste ca standard de baza intrucit acestea pot varia de la loc la loc prin factori relativi la traditie, considerente politice si socioeconomice, nu neaparat legate de stiinta medicala.

4. EVALUARI ALE MECANISMULUI DE INFERENTA

Sistemele expert cadru permit construirea seturilor de reguli de diverse complexitati, complexitatea fiind in general limitata de structura hardware. Nu toate cunostintele din domeniul de expertiza sint luate in consideratie cind o anumita premisa este indeplinita. Una dintre principalele caracteristici a sistemelor expert este ceea a determinarii acelor cunostinte ce sint relevante pentru situatia data. In limbajele de programare traditionale nu se pune problema determinarii instructiunii urmatoare de executat deoarece toate instructiunile sint executate consecutiv. O structura conditionala care foloseste instructiuni IF-THEN/ELSE, GOTO, bucle, este un caz extins de natura secventiala sau procedurala a limbajului de programare.

Procesorul de inferenta alege din informatiile furnizate de utilizator pe acelea ce sint aplicabile in rezolvarea problemei.

Exista doua strategii de control ce se pot folosi in determinarea regulii de selectie, inlantuirea inainte si inlantuirea inapoi. Pentru ilustrare se va considera un sistem expert bazat pe reguli de productie.

Inlantuirea inapoi este metoda de control cea mai utilizata in sistemele expert. In acest caz sistemul incepe cu scopul propus, apoi examineaza setul de reguli pentru a gasi prima regula care poate sa atinga scopul propus. Daca premiza primei reguli este TRUE, atunci sistemul realizeaza actiunea corespunzatoare. Daca premiza este FALSE, atunci procesorul de inferenta considera regula urmatoare pentru atingerea scopului. Daca premiza acesteia este TRUE atunci regula este executata. Utilizatorul poate alege inlantuirea inapoi in cazul in care procesorul de inferenta al sistemului cauta un scop. Gasirea valorii devine un scop nou (temporar) al programului. Prin inlantuire inapoi, pentru a determina informatia necesara, se cauta in lista de reguli o regula care atribuie o valoare pentru noul scop - de aici "inlantuire inapoi" - pina ce setul de reguli este epuizat. Se analizeaza premiza noii reguli conform necesitatilor curente. Daca valoarea de adevar a premizei nu poate fi stabilita, fiind necunoscuta, devine un nou scop si ciclul se repeta. Un sistem expert permite reluarea pas cu pas a procesului prin care procesorul de inferenta a ajuns la decizia sau solutia particulara. Directia de inlantuire, inainte sau inapoi, este cea mai importanta contributie la capacitatea sistemului expert de a raspunde intrebarilor puse.

Inlantuirea inainte porneste de la premiza la actiune pina cind variabilei scop i se atribuie o valoare. Procesorul de inferenta incepe prin a cauta in setul de reguli prima regula a carei premiza este TRUE. Aceasta va fi executata si intoarce un rezultat ce va putea fi folosit pentru testarea altor reguli. Aceasta valoare devine o parte din baza de date si este disponibila si pentru alte reguli. La urmatoarea trecere prin setul de reguli, se cauta prima regula a carei premiza este TRUE.

Prin mecanismul de inferenta se arata cum utilizeaza sistemul cunostintele si datele clinice pentru a construi rezultatul. Un mecanism de inferenta poate fi impartit in trei mari categorii dupa modul in care aceasta se realizeaza:

* inferenta bazata pe similaritate, incluzind date statistice.
* inferenta cauzala.
* inferenta deductiva bazata pe reguli explicite.

Mecanismul de inferenta bazat pe similaritate are mai ales o importanta istorica, primele sisteme pentru diagnoza medicala au fost pur empirice si statistice. La acestea frecventa simptomelor si semnalmentelor au fost tabelate alaturi de conditiile de adevar obtinute din observarea pacientilor. Aceasta abordare a fost motivata prin presupunerea ca daca s-a obtinut odata un succes se poate asuma riscul reproductibilitatii. In jurul anilor

Scanned with OKEN Scanner

1975 s-a dovedit ca aceasta abordare produce bune rezultate in domenii clinice inguste. Din acest motiv cele mai multe cercetari imbraca in sistemele expert reale combinatii intre factori statistici, conostinte teoretice si opinia expertului. Toate metodele statistice referitoare la noii pacienti pornesc de la presupunerea similaritatii. Metodele statistice difera dupa maniera in care se defineste similitudinea si deci, de modul in care se exploreaza datele din baza de date. Pot fi facute diferite presupuneri de distributie. Procedurile statistice acopera intre 5 si 50 % din cazurile similare si permit reducerea bazei de date pornind de la principiul similitudinii informatiilor. Aceste proceduri pot fi impartite in doua categorii, ambele pornind de la probabilitatea conditionala a profilului de sindromuri asociate pacientului. O clasa de modele pornesc de la probabilitatea directa, pe cind cealalta pastreaza descrierea statistica a simptomatologiei bolii, utilizind probabilitatea inversa calculata dupa formula inversa Bayes.

Mecanismele de inferenta cauzale utilizeaza declaratii cauzale calitative, (insulina KT determina modificari intracelulare) sau modele fiziologice si farmacocinetice. Cunostintele de acest tip sint reprezentate in baza de cunostinte si se numesc cunostinte 'profunde'. Sistemul consta dintr-o retea complexa de interconectare a regulilor de acest tip, regulile simple putind interactiona in diverse moduri, asa ca, prin inspectarea bazei de cunostinte fiecare regula poate sa aduca o contributie majora la rezultat. Retelele cauzale pot contine bucle prin faptul ca insasi mecanismul cauzal pe care il reprezinta implica mai multe reevaluari. Pot fi reprezentati astfel doi factori anatomici: adiacenta si conexiunile functionale.

Un exemplu in acest sens este NEUROLOGIST, sistem pentru localizarea placilor in scleroza multipla, in care o placa poate afecta traiectele auxiliare. Conexiunile anatomice determina legaturi false cum este cazul invaziilor cancerigene. Cu toate ca acestea utilizeaza cunostinte anatomice, o anatomie pe calculator nu a fost realizata inca.

5. ANALIZA CONSECINTELOR

In ultimul timp multe SSD au fost introduse in practica clinica obisnuita. Din mica experienta obtinuta se pot extrage concluzii privind implicatiile asupra pacientilor, personalului, administratiei, implicatii ce pot avea consecinte considerabile.

Cum principalul scop al SSD este obtinerea rapida si precisa a diagnosticului, intr-o evolutie normala utilizarea SSD determina o imbunatatire a tratarii pacientului. Astfel cunostintele expertului de inalt nivel si priceperea in judecarea clinica a specialistilor de la nivelele superioare este disponibila pentru pacientii tratati la toate nivelele sistemului de asistenta medicala. Calitatea unui SSD este limitata de utilizarea procedurilor standard. Pacientul poate observa ca procesul consultarii este inuman atunci cind medicul lucreaza cu calculatorul si nu se concentreaza pe relatiile personale si

Scanned with OKEN Scanner

consideratiile clinice. Daca calculatorul devine o parte integranta a vietii zilnice atitudinea se schimba, in principal prin faptul ca pacientii insisi lucreaza cu acesta. In acest sens sint disponibile studii de profunzime in Glady system (Lucas 77). O motivatie serioasa in utilizarea SSD este garantarea stricta a standardizarii principiilor de tratament.

Pentru medic orice formalism pus ca procedura de obtinerea diagnosticului are ca efect cresterea performantelor diagnosticarii. SSD poate asista doctorul in colectarea datelor relevante despre pacienti si produce un bun raport medical. Multe SSD cer informatii standardizate si formalizate ce determina studii prospective si retrospective fata de inregistrarile traditionale. Pot fi introduse mai intii astfel de sisteme pentru cresterea aderentei utilizatorilor prin facilitatile ce le ofera in studiile prospective si retrospective. In acest mod SSD poate fi un important instrument in cercetarea clinica din multe zone medicale (Kent 85). Functie de sistem si de motivarea atitudinii medicului SSD poate creste competenta utilizatorului sau reducerea calificarii sale. Daca sint folosite numai functiile critice sau de invatare, SSD, se constituie intr-un bun instrument de marire a competentei profesionale. Anumiti doctori privesc introducerea SSD in cadrul medical ca un mijloc de coordonare si supraveghere a activitatii lor clinice.

Spitalul poate avea o serie de beneficii prin utilizarea SSD. Un sistem bun poate scadea costurile departamentului dupa cum un sistem cu puncte slabe poate creste numarul testelor de laborator si a investigatiilor. Standardizarea terminologiei creste controlul calitatii cercetarilor clinice. Trebuie considerat si costul achizitiei hardwerului si a sistemului de programe, al scolarizarii personalului. Sint simplu de estimat influientele timpilor de raspuns, precum si a erorilor ce apar in momentul in care SSD este utilizat in mod curent. Daca SSD necesita pentru utilizare si mentinere personal calificat, sint necesare cheltuieli substantiale greu de recuperat. Introducerea SSD poate conduce la regindirea structurilor ierarhice traditionale in responsabilitatea deciziei medicale din spitale. O decizie SSD este mai greu de dezbatut decit o propunere din partea unui membru al colectivului. Daca SSD este utilizat de cadrela din nivelele ierarhice inferioare pot apare conflicte greu de estimat intre acestia si nivelele ierarhice superioare. De asemenea utilizarea SSD in practica medicala va determina a semnificativa alterare a structurii specialistilor. Teoretic utilizarea SSD in spitalele locale va determina reducerea preferintei pacientilor pentru spitalele regionale si universitare ce beneficiaza de personal medical cu inalta specializare.

O serie de aspecte sociale ale utilizarii SSD pe scara larga sint greu de estimat. Utilizarea extensiva a SSD poate determina schimbarea imaginii de zi cu zi a medicilor, multi pacienti vor cere sa participe efectiv la diagnosticare, doctorul fiind mai mult un translator si un comentator. Chiar daca SSD devine disponibil pentru utilizarea in public nu poate prescrie un diagnostic ca o alternativa a consultului medical. Profesiunea

Scanned with OKEN Scanner

medicala stricta accepta noile metode de decizie si multe reguli impuse de SSD prin diferite autoritati medicale nationale si internationale pot determina dezvoltari majore ale SSD, insa este necesara satisfacerea compromisului intre anarhie si sufocare.

Aspectele de natura etica trebuiesc si ele luate in consideratie. Astfel ca este neetic sa nu se foloseasca astfel de sisteme daca ele conduc la cresterea grijii de sanatate. Citeva conflicte etice uzuale in practica medicala pot fi ascunse prin consultarea utilizind SSD. Un exemplu este consultul genetic unde informatii biotehnice si statistice pot ocoli intrebari obscure si intime.

Desigur ca se nasc si o serie de alte intrebari in practica utilizarii SSD in medicina. Este stabilit principiul ca medicul este raspunzator de decizia pe care o ia fata de pacient. In general aceasta responsabilitate se extinde si asupra superiorilor, mai ales asupra sefului de departament. Extensia naturala a acestui principiu face ca decizia sugerata de SSD sa confere responsabilitatea in ultima instanta medicului ce a confirmat-o. Astfel, se aliniaza cu metodologia practica in care medicul poarta responsabilitatea evaluarii sugestiilor critice sau avizarii textelor din carti sau lucrari stiintifice.

Complexitatea SSD in continua crestere, determina implicatii si in alte zone profesionale, cunoasterea de catre medici a unor notiuni legate de stiinta calculatoarelor, cercetari teoretice ale teoriei probabilitatii, multimilor vagi, cit si implicatii provenind din schimbarile survenite in structura sistemelor expert. O practica medicala competenta in viitoarele sisteme expert medicale este extinsa la includerea competentelor de intelegere, certificare si testare. Pe de alta parte erorile ce pot apare la sistemele computerizate pot cauza multe suferinte, motiv pentru care o legislatie speciala trebuie sa reglementeze acest lucru. Experienta de mai multi ani in utilizarea SSD arata ca duce la cresterea obligatiilor utilizatorului.

6. CONCLUZII

Desigur ca, pnctele de vedere exprimate aici se bazeaza pe informatiile cuprinse in literatura de specialitate corelate cu observatii proprii ale autorului in urma discutiilor cu personal avizat din domeniul medicinei. Nu au fost tratate detaliile tehnice de constructie a sistemelor suport de decizie cu aplicatii medicale, ci doar problematica generala a aplicatiilor cu astfel de destinatii.

7. BIBLIOGRAFIE

1. Teza de doctorat, D.Carstoiu 1989.
2. Fuzzy Set Theory and its Aplications, H. J. Zimmermann, 1988.
3. Readings in medical artificial intelligence, W. J. Clancey, E. H. Shortliffe, 1990.
4. Logics for artificial intelligence, R. Turner, 1985.

Scanned with OKEN Scanner

ANALIZA NUMERICA A CIMPULUI MAGNETIC DE EXCITATIE LA UN VIBRATOR ELECTRODINAMIC

Ioan Vadan, Stelian Perde, Anca Farcas
Institutul Politehnic Cluj-Napoca

Rezumat

Se prezinta o analiza numerica a cimpului magnetic de excitatie la un vibrator electrodinamic, utilizind metoda elementului finit implementata pe un calculator PC compatibil IBM, folosind limbajul FORTRAN 77. Programul se poate folosi pentru calculul oricarei configuratii de cimp magnetic stationar plan-paralel sau plan-meridian.

1. Introducere

In Figura 1 se prezinta o sectiune printr-un astfel de vibrator electrodinamic. Principiul lui de functionare este asemanator unui difuzor electrodinamic. Bobina de excitatie (7) produce un cimp magnetic constant in intrefierul al circuitului magnetic (5). Aici se afla bobina mobila (4), prin care trece un curent alternativ sinusoidal, care, prin interactiune cu cimpul magnetic constant din intrefier va produce o forta alternativa si sinusoidala care va produce vibrarea echipamentului mobil suspendat de sistemul elastic (2), supunind o eventuala piesa fixata pe platanul (6) la vibratii controlate.

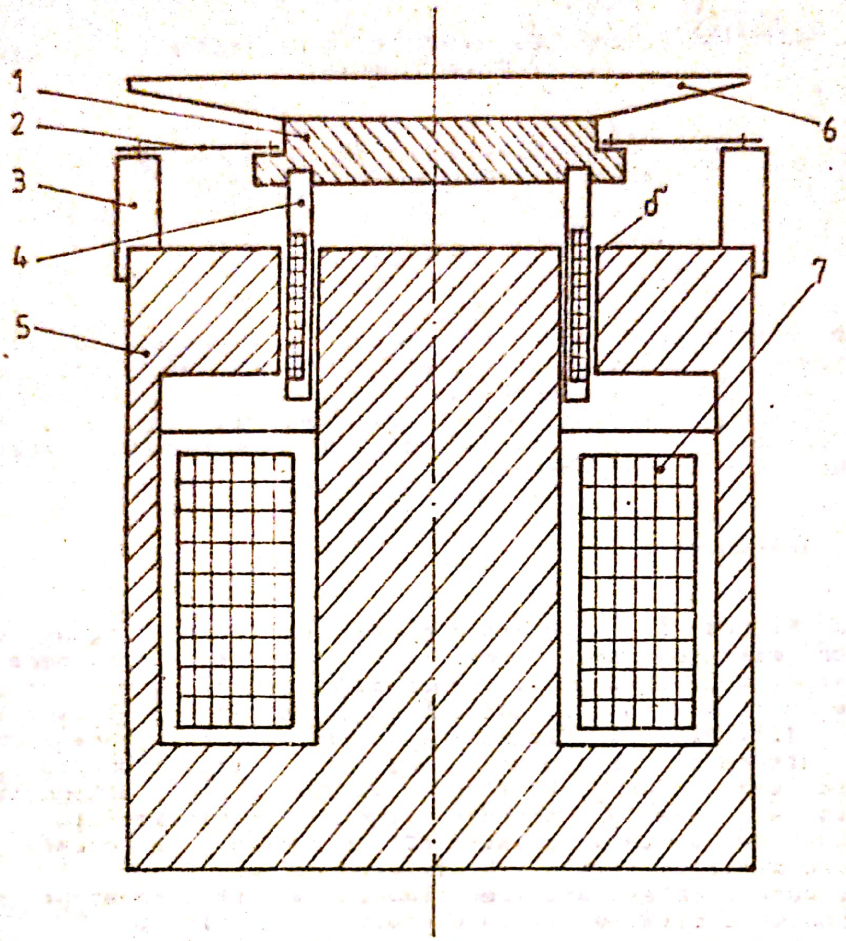
Se pune problema analizei cimpului magnetic constant produs de bobina de excitatie (7) in circuitul magnetic (5).

In prezent, calculul cimpului magnetic la o astfel de constructie de circuit magnetic se realizeaza prin metode numerice [1] in conditii foarte avantajoase, deoarece circuitul magnetic prezinta simetrie cilindrica si calculul cimpului se poate reduce la doua dimensiuni intr-un plan meridian care taie vibratorul prin axa sa.

In lucrare se prezinta solutia aleasa de autori pentru analiza numerica a cimpului magnetic de excitatie la un astfel de vibrator electrodinamic.

S-a adoptat metoda de analiza prin element finit. S-a adoptat solutia elaboratii unui program propriu de analiza numerica a cimpului magnetic fata de posibilitatea achizitionarii unui set de programe de institutii specializate (ex. I.C.P.E. Bucuresti), atit din considerente economice, cit si din posibilitatea creata in acest fel de a dezvolta aceste programe pina la integrarea lor intr-un program complex de proiectare optimizata asistata de calculator a vibratoarelor electrodinamice.

Scanned with OKEN Scanner



1. Piesă fixare platan bobină mobilă
2. Sistem elastic
3. Suport sistem elastic
4. Bobină mobilă
5. Circuit magnetic
6. Platan
7. Bobină fixă de excitație

Fig. 1.

Scanned with OKEN Scanner

S-a plecat de la implementari clasice ale metodei elementului finit prin programarea in FORTRAN si rularea pe calculatorul FELIX C-256 utilizind pentru rezolvarea sistemului de ecuatii algebrice subprogramul REBAN din biblioteca matematica MATHLIB.

S-a implementat apoi acest program pe un calculator compatibil IBM-PC AT, utilizind compilatorul F77.

Programul, PROF01, este deci scris in limbajul FORTRAN si foloseste date din fisierul FOR01.

In continuare se vor prezenta metoda, programul intocmit si rezultatele obtinute.

2. Metoda elementului finit

In Figura 2 se prezinta discretizarea obtinuta pentru domeniul de calcul ales, domeniu care acopera o semisectiune plan meridiana prin vibrator si un spatiu in aer deasupra intrefierului.

S-au obtinut un numar de 340 de elemente si 198 de noduri. De asemenea, un numar de 48 de elemente sint cu surse, se suprapun peste zona din fereastra circuitului magnetic ocupata de bobina de excitatie. S-a folosit pentru caracterizarea cimpului magnetic potentialul magnetic vector A, care in acest caz cu simetrie cilindrica are doar o singura componenta $A_\varphi = A$.

Considerind cazul cimpului magnetic stationar, in medii fixe, liniare, izotrope, neomogene si nemagnetizate permanent, modelul matematic va fi o ecuatie cu derivate partiale de tip Poisson :

$$\nabla (\nu_a \nabla \bar{A}) = - \bar{J} \qquad (1)$$

cu : $\bar{A}$ - potentialul magnetic vector ;
$\bar{J}$ - vectorul densitatii de curent.

Intr-un cimp bidimensional plan-paralel, ecuatia (1) ia forma :

$$\frac{\partial}{\partial x} \left(\nu_a \frac{\partial A}{\partial x} \right) + \frac{\partial}{\partial y} \left(\nu_a \frac{\partial A}{\partial y} \right) = - J \qquad (2)$$

Intr-un cimp bidimensional plan-meridian, ecuatia (1) ia forma :

$$\frac{\partial}{\partial r} \left[\frac{\nu_a}{r} \frac{\partial}{\partial r} (rA) \right] + \frac{\partial}{\partial z} \left(\nu_a \frac{\partial A}{\partial z} \right) = - J$$

Daca se noteaza cu $\Psi = r A$, o asa-numita functie de flux, atunci ecuatia (3) devine

$$\frac{\partial}{\partial r} \left(\frac{\nu_a}{r} \frac{\partial \Psi}{\partial r} \right) + \frac{\partial}{\partial z} \left(\frac{\nu_a}{r} \frac{\partial \Psi}{\partial z} \right) = - J$$

Scanned with OKEN Scanner

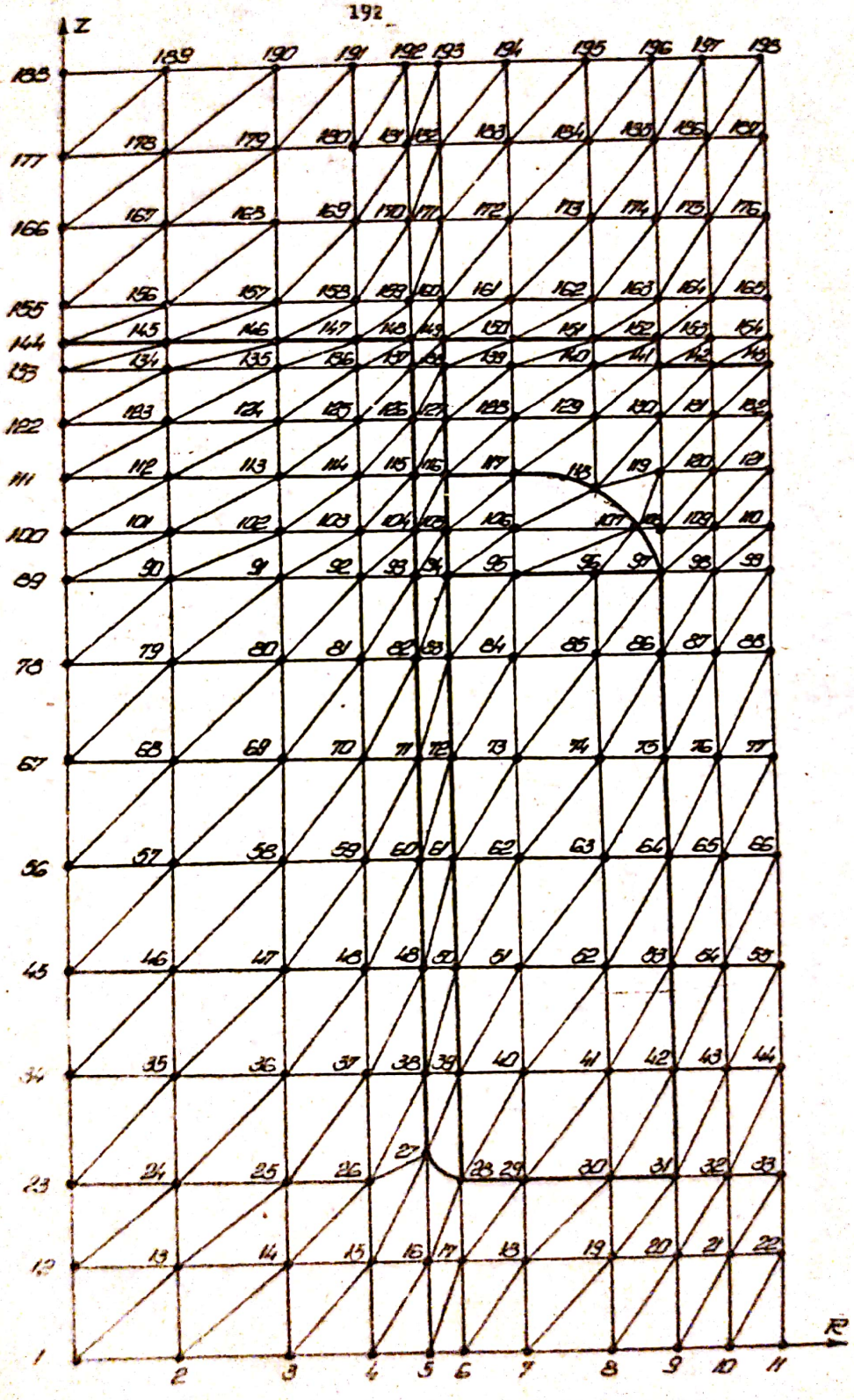


fig.2

E16

Scanned with OKEN Scanner

Se observa similitudinea dintre relatia (2) si (4), ceea ce ne permite sa folosim acelasi program, atit pentru analiza cimpurilor magnetice cu simetrie plan-paralela, cit si cu simetrie plan-meridiana prin trecerea de la coordonatele x,y la r,z si de la A la Ψ cu modificari minimale.

Noi consideram cimpul descris de ecuatia (4).

Conditiile pe frontiera domeniului le alegem de tip Dirichlet, si anume $\Psi|_{\Gamma} = 0$, adica consideram potentialul magnetic vector pe frontiera nul, si prin urmare va fi nul si potentialul magnetic vector modificat $\Psi = r A$.

Considerind o formulare variationala a problemei de cimp descrise de ecuatia (4), se demonstreaza in [1] ca rezolvarea ei este echivalenta cu minimizarea functionalei energetice :

$$F(\Psi) = \int_{\Omega} \left\{ \frac{\nu_a}{2r} \left[\left(\frac{\partial \Psi}{\partial z} \right)^2 + \left(\frac{\partial \Psi}{\partial r} \right)^2 \right] - J \cdot \Psi \right\} dz\, dr - \int_{\Gamma} \frac{\nu_a}{r} \frac{\partial \Psi}{\partial n} \Psi \, d\Gamma \qquad (5)$$

unde Γ este frontiera domeniului Ω.

La nivelul fiecarui element e se aproximeaza functia Ψ prin :

$$\Psi^e(r,z) = \sum_{i=1}^{3} \varphi_i \Psi_i = [\phi]^e \cdot [\Psi]^e \qquad (6)$$

unde functiile φ_i au expresia :

$$\varphi_i = \frac{a_i + b_i \cdot z + c_i \cdot r}{2S^e} \qquad (7)$$

iar :

$$a_i = r_j \cdot z_k - r_k \cdot z_j \qquad (8)$$

$$b_i = z_j - z_k \qquad (9)$$

$$c_i = r_k - r_j \qquad (10)$$

iar a_j , a_k , b_j , b_k , c_j , c_k au expresii analoage.

Conditia de stationaritate pentru functionala (5) se scrie :

$$\partial F / \partial \Psi_i = 0$$

adica la nivelul unui element :

$$\int_{\Omega^e} \frac{\partial}{\partial \Psi_i} \left\{ \frac{\nu_a}{2r} \left[\left(\frac{\partial \Psi_e}{\partial z} \right)^2 + \left(\frac{\partial \Psi_e}{\partial r} \right)^2 \right] dr\, dz - \right.$$

$$\left. - \int_{\Omega^e} J \frac{\partial \Psi_e}{\partial \Psi_i} dr\, dz - \int_{\Gamma^e} \left(\frac{\nu_a}{r} \frac{\partial \Psi_e}{\partial r} \right) \frac{\partial \Psi_e}{\partial \Psi_i} d\Gamma^e = 0 \right. \qquad (11)$$

In ipoteza ca pe un element

$$\nu = \nu^e = \text{const.} \quad \text{si} \quad r = r_e = \frac{1}{3} \sum r^e = \text{const.}$$

primul termen din ecuatia (11) devine

$$\frac{\nu^e}{r_e}\iint_{\Omega e}\left(\frac{\partial\psi_i}{\partial z}\cdot\left[\frac{\partial\phi}{\partial z}\right]^e\cdot\{\psi\}^e+\frac{\partial\psi_i}{\partial r}\left[\frac{d\phi}{dr}\right]^e\cdot\{\psi\}^e\right)dr\,dz=$$

$$=\frac{\nu^e}{r_e(4\cdot S_e)}\sum_{j=1}^{3}(b_i b_2+c_i c_2)\psi_j=\sum_{j=1}^{3}K_j\psi_j \qquad (12)$$

In mod similar, considerind $J=J^e=\text{const}$, termenul al doilea din ecuatia (11) devine :

$$\iint_{\Omega e}J\frac{\partial\psi^e}{\partial\psi_i}\,dr\,dz=J_e\iint_{\Omega e}\psi_i\,dr\,dz=\frac{J^e\cdot S^e}{3} \qquad (13)$$

Termenul al treilea din ecuatia (11) in cazul nostru este oul, deoarece nu avem conditii Neumann neomogene pe frontiera $\partial\psi/\partial n\neq 0$

Ca urmare ecuatia (11) se scrie matricial

$$K^e\cdot\{\psi\}^e-\{F\}^e=0 \qquad (14)$$

cu matricile $K^e(3\times3)$ si $F^e(3\times1)$ avind componentele

$$k_{ij}^e=\frac{\nu^e}{r_e(4S_e)}(b_i b_j+c_i c_j) \qquad (15)$$

$$f_i^e=\frac{J^e S^e}{3} \qquad (16)$$

Scriind ecuatiile pentru toate triunghiurile domeniului Ω rezulta un sistem de forma :

$$K\{\psi\}-\{F\}=0 \qquad (17)$$

unde necunoscute sint valorile ψ in nodurile retelei de discretizare.

Acesta este un sistem de ecuatii algebrice cu proprietatea ca matricea coeficientilor k este putin umpluta si are o structura de banda.

Acest sistem se poate rezolva cu ajutorul subprogramului REBAN din biblioteca matematica MATHLIB a limbajului FORTRAN. Acest subprogram utilizeaza metoda eliminarii a lui GAUSS pentru rezolvarea sistemelor de ecuatii cu matricea coeficientilor in forma de banda in jurul diagonalei principale.

Daca se tine cont ca aceasta matrice este si simetrica fata de prima diagonala, atunci se poate utiliza pentru rezolvare metoda radacinii patrate sau de eliminare a lui CHOLESKI.

Aceasta metoda este utilizata in programul ce va fi prezentat in continuare.

Matricea K se pune sub forma produsului a doua matrici triunghiulare : $K=L\cdot L^T$ unde

$$L=\begin{bmatrix}l_{11}&0&\dots&0\\l_{21}&l_{22}&\dots&\\\dots&\dots&\dots&\\l_{n1}&l_{n2}&\dots&l_{nn}\end{bmatrix};\qquad L^T=\begin{bmatrix}l_{11}&l_{12}&\dots&l_{1n}\\0&l_{22}&\dots&l_{2n}\\\dots&\dots&\dots&\dots\\0&0&\dots&l_{nn}\end{bmatrix}$$

iar $l_{ij}=l_{ji}$ prin relatiile :

$$l_{ii} = \sqrt{K_{ii} - \sum_{p=1}^{i-1} l_{pi}^2} \; ; \; l_{ij} = \frac{K_{ij} - \sum_{p=1}^{i-1} l_{pi} l_{pj}}{l_{ii}} \qquad (18)$$

Ecuatia $K \cdot \Psi = F$ devine $L \cdot L^T \cdot \Psi = F$ si notind $L^T \cdot \Psi = Y$ obtinem pe Y rezolvind ecuatia $L \cdot Y = F$ si apoi pe Ψ rezolvind ecuatia $L^T \cdot \Psi = Y$, ceea ce se face usor deoarece matricile L si L^T sint triunghiulare.

Se va obtine

$$y_i = \frac{F_i - \sum_{p=1}^{i-1} l_{ip} y_p}{l_{ii}} \; ; \; i = 1,2,\ldots,n \qquad (19)$$

si apoi

$$\Psi_i = \frac{y_i - \sum_{p=i+1}^{n} l_{ip} \Psi_p}{l_{ii}} \; ; \; i = n, n-1, \ldots, 1 \qquad (20)$$

la $i = n$ se ia $\Psi_n = y_n / l_{nn}$

Dupa rezolvare s-au calculat caracteristicile diferentiale ale cimpului, si anume componentele inductiei magnetice cu relatiile :

$$B_r = -\frac{\partial A}{\partial z} = -\frac{1}{r}\frac{\partial \Psi}{\partial z} \qquad (21)$$

si

$$B_z = \frac{1}{r}\frac{\partial}{\partial r}(Ar) = \frac{1}{r}\frac{\partial \Psi}{\partial z} \qquad (22)$$

3. Programul de calcul a cimpului

In Figura 3 se prezinta organigrama programului de calcul a cimpului magnetic. Acest program se compune din 7 blocuri distincte.

DATE INTRARE, asigura introducerea datelor de intrare cu variabilele mai importante : XT(198), YT(198) coordonatele nodurilor ;

LNODS(340,4) - o matrice care memoreaza la fiecare element nodurile alaturate lui in sens invers trigonometric si cifra 0 daca elementul este in aer si 3 daca elementul este in fier ; ea este echivalenta cu multimea MNODS(1360) ;

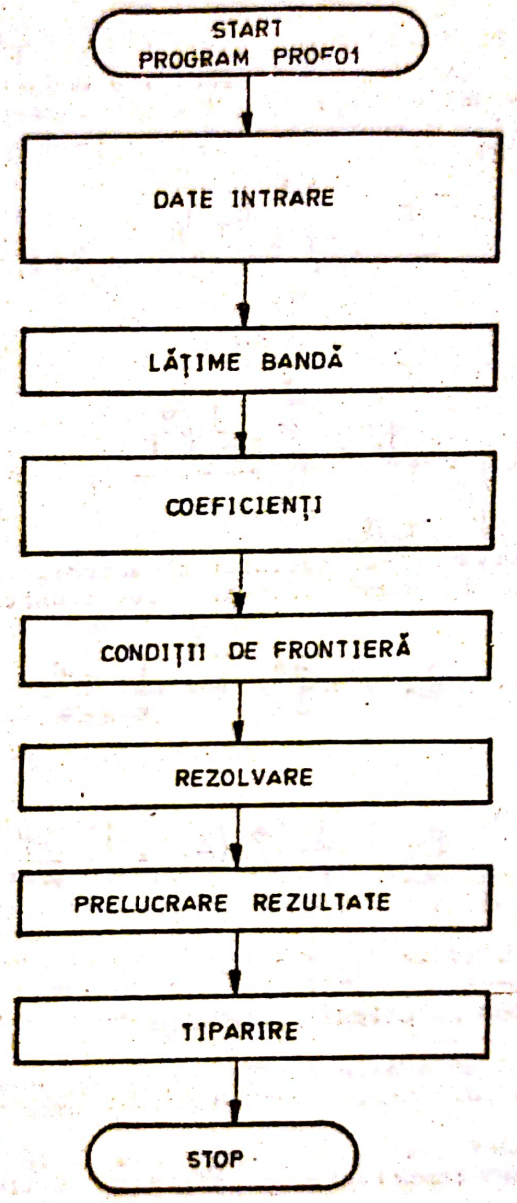


Fig. 3.

E 16

Scanned with OKEN Scanner

ESRES(3,3), AK(3), BK(3), CK(3) memoreaza caracteristici geometrice ale elementelor ;
NELEM si MELEM reprezinta numarul de elemente ;
NPOIN reprezinta numarul de noduri ;
NVFIX reprezinta numarul de noduri cu potential fix ;
NPROP reprezinta numarul de elemente cu surse ;
LVFIX(54) este lista nodurilor cu potential fixat ;
PVFIX(54) reprezinta potentialele fixe ale nodurilor ;
PROPES(48) sint densitatile de curent in elementele cu surse
NPROP1(48) reprezinta lista elementelor cu surse ;
ASRES(4500) este multimea elementelor matricei coeficientilor ;
ASTOK(198) este matricea coloana a termenilor liberi.

Blocul LATIME BANDA calculeaza latimea benzii si semibenzii matricei coeficientilor.

Blocul COEFICIENTI calculeaza elementele matricei coeficientilor K si a matricei termenilor liberi a ecuatiei algebrice obtinute dupa formulele deduse la paragraful 2.

Blocul CONDITII DE FRONTIERA asigura modificarea matricei coeficientilor in conformitate cu conditiile de frontiera date. In cazul nostru, la nodurile aflate pe frontiera domeniilor, deoarece potentialul este nul, se vor egala cu 0 toti coeficientii de pe coloana i si linia i care corespund nodului respectiv din matricea K, cu exceptia lui Kii, precum si termenul liber corespunzator, f_i .

Blocul REZOLVARE asigura rezolvarea sistemului de ecuatii algebrice prin metoda radacinii patrate.

Blocul PRELUCRARE REZULTATE asigura obtinerea din potentialele nodurilor a componentelor inductiei magnetice Br si Bz pe fiecare element.

Blocul TIPARIRE asigura tiparirea acestor rezultate.

4. Concluzii

Acest program de.calcul de cimp permite calculul cimpului magnetic plan-meridian si cu mici modificari si a celui plan-paralel. Discretizarea domeniului de calcul se face manual si rezultatele se dau sub forma unor liste ale potentialelor magnetice A in noduri si ale inductiei magnetice si a componentelor sale in fiecare element. Domeniul de calcul prezentat in Figura 2 este luat dupa un vibrator electrodinamic existent si s-au facut masuratori cu Teslametru a inductiei magnetice in intrefier la curenti prin bobina de excitatie egali cu cei considerati in datele din program, obtinindu-se o suprapunere foarte buna a datelor masurate peste cele calculate. In plus, s-a putut constata o corespondenta perfecta a semnului componentelor inductiilor magnetice in diferite zone ale circuitului magnetic, in totala concordanta cu cele presupuse teoretic.

Noi consideram acest rezultat deosebit de incurajator si in viitor dorim sa perfectionam acest program, prin completarea lui cu un subprogram de discretizare automata a domeniului de calcul, precum si cu un subprogram pentru trasarea (desenarea) liniilor de cimp. De asemenea, intentionam sa transcriem programul intr-un limbaj de programare mai evoluat, TURBO PASCAL sau TURBO C.

Scanned with OKEN Scanner

BIBLIOGRAFIE

1. Mindru, Gh., Radulescu, M.M. Analiza numerica a cimpului electromagnetic. Editura "Dacia" Cluj-Napoca, 1986.
2. Demircian, K.S., Ceciurin, V.L., Masinie rasceti electromagnitnih polei. Moskva, Vissaia Scola, 1986.
3. Cirstea, I., Pasare, S., Un model hibrid pentru cimpul magnetic stationar bidimensional, in E.E.A. Electrotehnica, nr.7/1989.

Metodă de apreciere a comportării întreruptoarelor de înaltă tensiune la comutaţie cu ajutorul calculatorului electronic

Ing. Curcanu George - I.C.M.E.T-Craiova
Prof.dr.ing.Floriganţă-Gh.-Univ.din Craiova
Ing.Defta Mihail- I.C.M.E.T.-Craiova

1. Introducere

Aprecierea comportării întreruptoarelor de înaltă tensiune la comutaţia curenţilor capacitivi şi stabilirea posibilităţilor limită de comutaţie a curenţilor de scurtcircuit se face prin determinarea caracteristicii dielectrice de gol ce a dat rezultate foarte bune pentru întreruptoarele cu aer comprimat şi hexafluorură de sulf (1).

Pentru întreruptoarele cu ulei puţin problemele sînt mai complexe, deoarece în acest caz există variaţii importante ale tensiunii de străpungere, datorate condiţiilor mediului de stingere în intervalul creat prin deschiderea contactelor, cît şi cele datorate fenomenelor statistice care apar în cazul străpungerilor dielectrice.

In acest caz se determină utilizînd calculatorul electronic, funcţia de comportarea întreruptorului pe baza modelului matematic al rigidităţii dielectrice a intervalului întreruptorului.

2. Experimentări pentru determinarea caracteristicii dielectrice de gol.

Cu instalaţia realizată în Laboratorul de Mare Putere al ICSITMTAE- Craiova s-au determinat tensiunile de străpungere între contactele în curs de deschidere ale întreruptorului cu ulei puţin tip UN - 12 KV/125o A, pe baza cărora se trasează caracteristica dielectrică de gol.

Schema de principiu a instalaţiei este prezentată în fig.1.

Pe polul T al întreruptorului cu ulei puţin tip UN-12 KV/125o s-a aplicat tensiunea continuă de la 5o KV în creştere cu 5 KV pînă la loo KV. S-au efectuat mai multe încercări, astfel că pentru fiecare treaptă a traductorului s-au determinat cel puţin 2o tensiuni de străpungere. Pentru fiecare treaptă a traductorului se consideră parametrii constanţi ai intervalului de rupere. Repetînd pe un număr mare de ori în condiţii identice măsurile tensiunii de străpungere, se constată, în general, că rezultatele aleatorii măsurării sînt caracterizate de o lege de repartiţie bine determinată. In acest caz se poate determina valoarea medie μ şi dispersia σ^2

./.

acestei repartiţii normale (Gauss) [1,2].

Fie $\mu_1, \ldots \mu_m$ tensiunile de străpungere corespunzătoare unei trepte a traductorului. In acest caz:

$$\mu = \frac{\sum_1^m u_j}{m} \quad ; \quad \sigma = \sqrt{\frac{\sum u_j - \mu}{m-1}} \qquad (1)$$

cunoscîndu-se aceste date, rezultă funcţia de comportare [2]:

$$S_i(u) = \frac{1}{\sigma_i \sqrt{2\pi}} \int_{-\infty}^{u} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}} \cdot dt \qquad (2)$$

Pentru comportarea întregului întreruptor se scrie funcţia cumulativă:

$$S(u) = \sum_{j=1}^{K} \prod_{i=1}^{j} [1 - S_{i-1}(u)] \cdot S_i(u) \qquad (3)$$

care determină probabilitatea de străpungere în timpul intervalului K sau în cursul intervalului precedent.

K - reprezintă numărul de trepte ale traductorului de cursă

3. Programe realizate pe calculator pentru determinarea probabilităţii de străpungere

Pe calculatorul Independent I-102 F din dotarea Laboratorului de Mare Putere, s-au realizat două programe de calcul în limbaj FORTRAN pentru determinarea probabilităţii de străpungere a întreruptorului între contactele în curs de deschidere ale întreruptorului cu ulei puţin.

Programul a fost realizat pentru :

- verificarea normalităţii şirului de date experimentale;
- calculul valorii medii şi al dispersiei

Programul II a fost realizat pentru:

- calculul funcţiei de comportare: K . Si;
- calculul funcţiei cumulative: S(u).

Programul I

Există un număr m de date experimentale care reprezintă tensiunile u_j, de străpungere a intervalului, obţinute în urma experimentărilor efectuate cu instalaţia laboratorului.

Măsurătorile au fost efectuate cu instalaţia laboratorului repetate de mai multe ori şi s-a constatat că rezultatele aleatorii u_j, sînt caracterizate de o lege de repartiţie normală.

In acest scop s-a verificat normalitatea şirului de date experimentale pentru fiecare treaptă la care s-a iniţiat străpungerea. Verificarea normalităţii şirului de date experimentale s-a făcut pe baza metodei diferenţelor succesive a lui B-Abbe (6).

Testul lui E.Abbe constă în următoarele:

- se introduc datele într-un fişier pe disc;
- se ordonează datele şirului în ordine crescătoare:

$u1 < u2 < \ldots < u_n$;

- se calculează valoarea medie μ şi dispersia σ cu relaţiile:

$$\mu = \frac{1}{m}\sum_{1}^{m} u_j \quad ; \quad \sigma = \sqrt{\frac{\sum_{1}^{m}(u_j-\mu)^2}{m-1}} \qquad (4)$$

- se calculează raportul "γ" cu relaţia:

$$\gamma = \frac{\sum_{1}^{m}(u_{j+1}-u_j)^2}{\sum_{1}^{m}(u_j-\mu)^2} \qquad (5)$$

- se alege pragul de semnificaţie $q = 0{,}05$;
- se calculează γ_q cu relaţia:

$$\gamma_q = 1 - \frac{1{,}645\sqrt{m-2}}{\sqrt{(m-1)(m+1)}} \qquad (6)$$

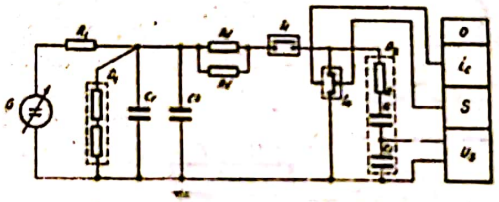


Fig. 1. Schema de principiu a instalaţiei pentru determinarea tensiunii de străpungere a intervalului întreruptorului; G_c — sursă de tensiune continuă variabilă; R_1 — rezistenţă de protecţia sursei; D_1 — divizor rezistiv pentru măsurarea tensiunii sursei; C_1; C_2 — condensatoare de înaltă tensiune; R_d — rezistenţă de descărcare; I_1 — întreruptor auxiliar; I_u — întreruptor de încercat; D_2 — divizor capacitiv pentru măsurarea tensiunii de străpungere; C_1; C_2 — condensatoare de înaltă şi joasă tensiune; O — osciloscop catodic; I_c — curent comandă aparat; S — cursă traductor; U_s — tensiune de străpungere.

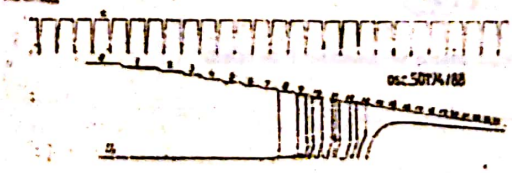


Fig. 2. Oscilograma tensiunii de străpungere între contactele în curs de deschidere ale întreruptorului tip UN-12 kV/1 250 A.

- unde m reprezintă numărul de date experimentale.

Dacă se constată că $\beta_1 < \beta_q$, se admite ipoteza normalității șirului de date. Organigrama programului I este dată în fig.3.

Conform acestui program rezultă normalitatea șirului de date.

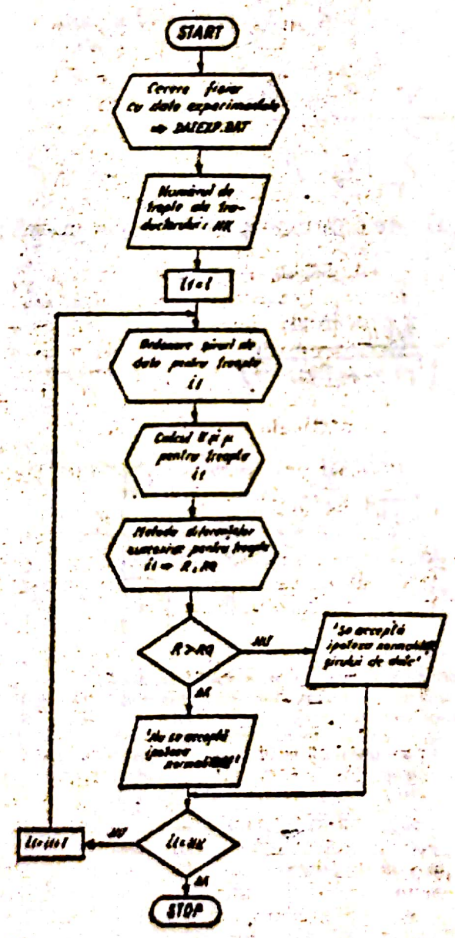


Fig. 3. Organigrama programului I de calcul.

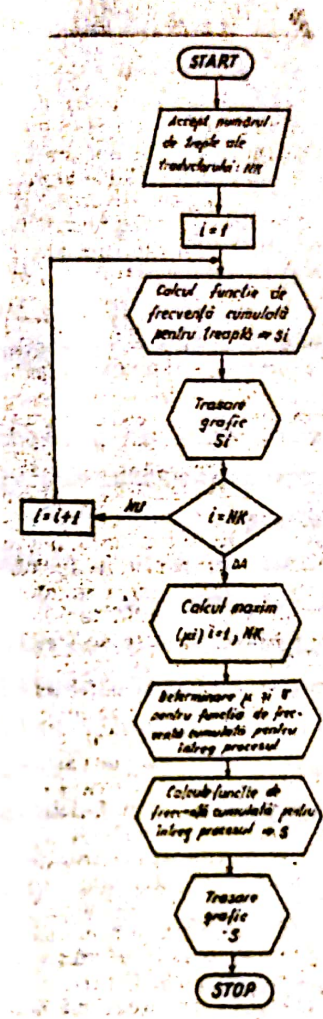


Fig. 4. Organigrama programului II de calcul

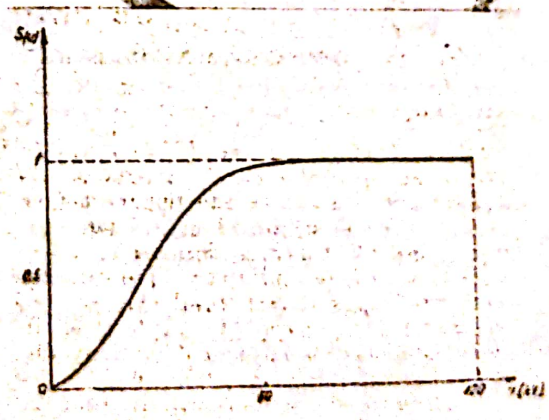


Fig. 5. Graficul funcției cumulative trasată pe calculator.

Scanned with OKEN Scanner

Programul II

Se calculează K.S_i-funcţie de comportare şi se trasează grafic curbele funcţiei de comportare pentru fiecare treaptă.

S-a calculat de asemenea, funcţia cumulativă S(u), care dă probabilitatea de străpungere a intervalului şi s-a trasat curba funcţiei cumulative. Organigrama programului de calcul este prezentă în fig.4, iar graficul funcţiei cumulative este dat în fig.5.

4 Concluzii

Pentru aprecierea comportării întreruptoarelor de medie tensiune cu ulei puţin la probele de comutaţie se procedează astfel:

- se determină tensiunile de străpungere ale intervalului dintre contactele în curs de deschidere ale întreruptorului în stare nouă de funcţionare;
- se ordonează şirul de date experimentale şi se verifică dacă se supun unei repartiţii normale;
- se determină funcţia de comportare a întreruptorului utilizînd modelul matematic al rigidităţii dielectrice a intervalului întreruptorului.

Pe baza acestei metode se poate estima mai exact probabilitatea de defect a întreruptorului, decît prin trasarea directă a caracteristicii de gol a întreruptorului.

Bibliografie

1. Frohlich, K. Mesure des caracteristiques de tension et performances de coupure des disjoncteurs. In: CIGRE-13.01.1978
2. Eberenberger, H. Statistische Verfahren zur Bestimmung der Durchschlagwahrscheinlichkeit von Isolierstreken. In: Wiss Z.Elektrotech. Leipzig 17, 1971, 2/3 S.- 11713-2.
3. Curcanu, C.; Barinca, O; Burlacu, I. Equipment for determination of the voltage circuit-breaker dielectric rigidity characteristic. In: Technical reports-Electroputere Craiova-1986
4. Curcanu, C., Barinca, O, Defta, M. Rassuzhdenia nad harakteristikami holostogo hoda visokovoltnih vicliuchatelej. Conferinta de Aparataj Electric- INTERELEKTRO, Togliatti, URSS, 12-17-09-1988.
5. Curcanu, C., Defta, M, Muller, M Vozmojnosti ispolzovania dielektricheskih karakteristik holostogj hoda pri prognozirovanii povedenia malogabiomnih masliannih vikliutchatelei na srednem napriazhenii. In:Referatele Conferinţei Jubiliare a IET-Sofia, 27-28-10-1988.
6. Tiron, I. Teoria erorilor de măsurare şi metoda celor mai mici pătrate. Editura Tehnică, Bucureşti 1972.

PROGRAM PENTRU CALCULUL NECESARULUI DE CALDURA PENTRU CONSTRUCTII CIVILE SI INDUSTRIALE

autori: conf.dr.ing.Claudia Botez,I.P.Iaşi
ing.Virgil Botez, Camera de Comerţ Iaşi

Definirea necesarului de căldură pentru clădirile civile şi industriale este o etapă premergătoare proiectării şi dimensionării instalaţiilor de încălzire.

Procedura de calcul a necesarului de căldură este reglementată de prevederile STAS 1907/ 1 şi 2 din 1980,STAS 6472/ 1,2 şi 3 din 1989 completate de instrucţiunile 41 din 18.12.89 ale ICCPDC cu privire la asigurarea stabilităţii termice.

Lucrarea îşi propune să prezinte o procedură de calcul a necesarului de căldură,elaborată pentru calculatoare personale compatibile IBM - PC.

Procedura nu se aplică la: încăperi subterane,clădiri climatizate,clădiri cu destinaţii speciale.

1. Prezentare generală

In fig.1 este dată schema generală de calcul din care se desprind două etape de lucru:

- calculul coeficienţilor de masivitate termică al elementelor de construcţie (conform STAS 6472/3 din 1989)
- calculul necesarului de căldură (conform STAS 1907/1 din 1980).

2. Descrierea procedurii de calcul

Coeficienţii de masivitate termică sînt calculaţi după relaţiile indicate în [4] pentru fiecare element de construcţie în parte.Aceşti coeficienţi se depun în tabele care trebuiesc actualizate pentru fiecare construcţie-obiectiv în parte.

Pentru calculul necesarului de căldură datele de intrare privesc construcţia în ansamblu,încăperea şi elementele de construcţie ale unei încăperi.

Necesarul de căldură corespunde pierderilor fiecărui

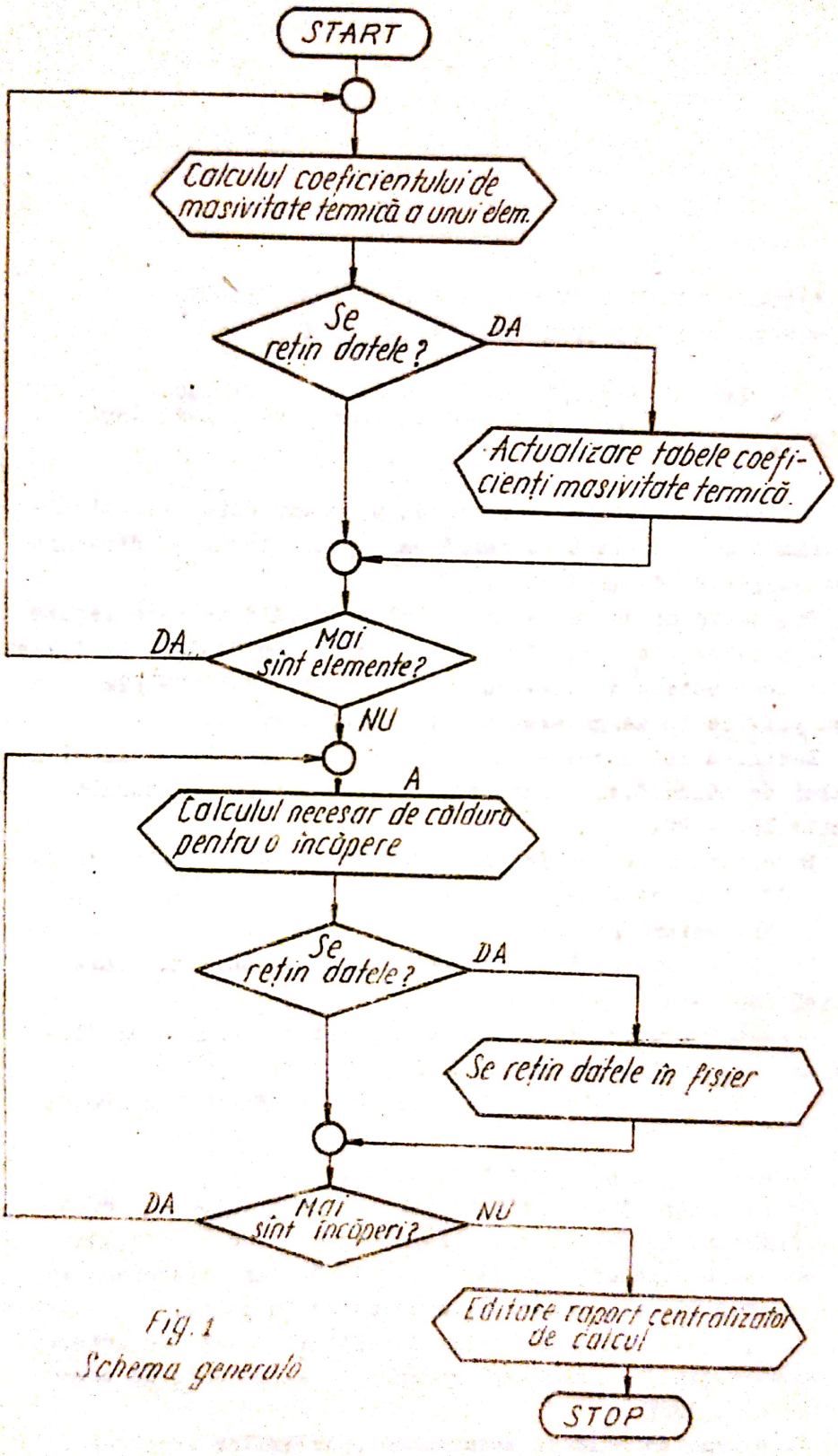


Fig. 1
Schema generală

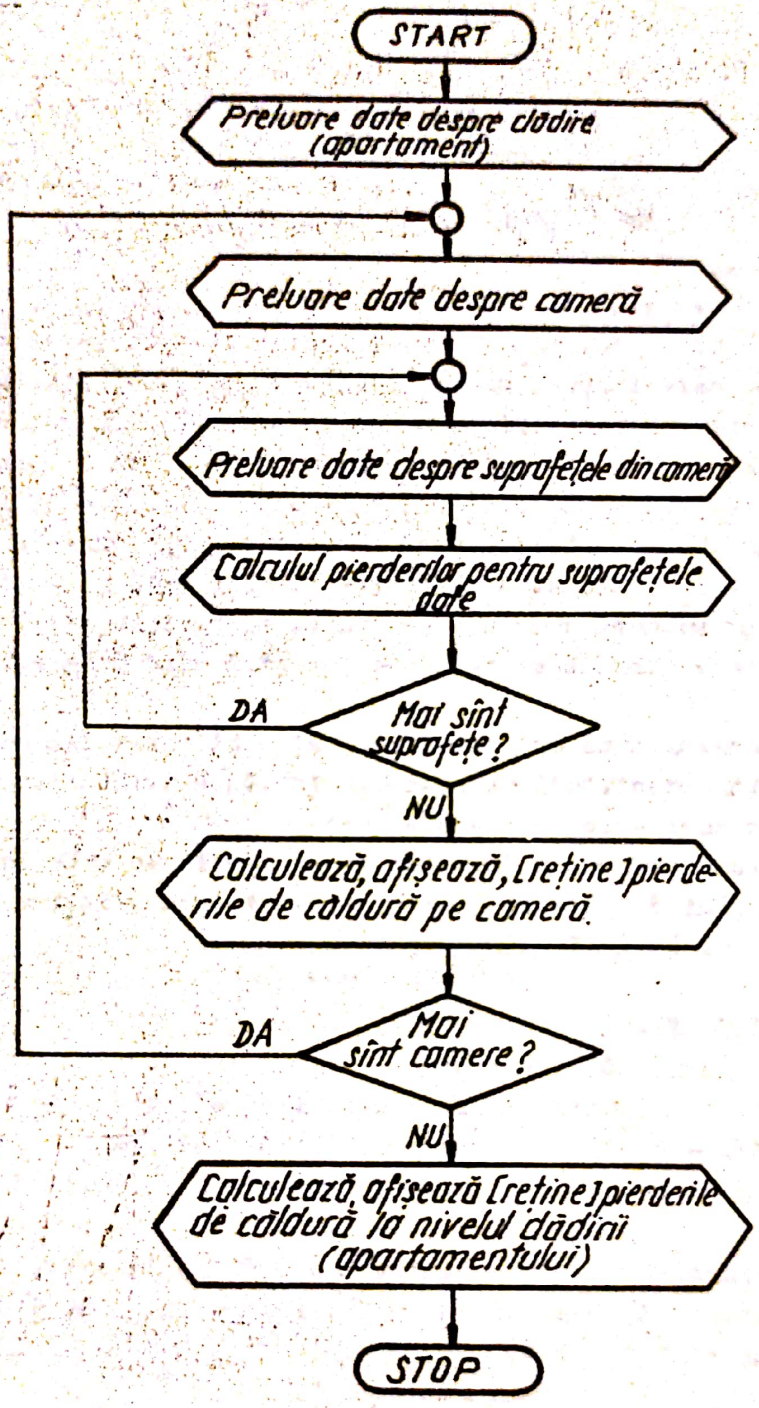


Fig. 2
Calculul necesarului de căldură pentru o clădire (apartament)

Scanned with OKEN Scanner

fiecărui element de construcţie,cumulat la nivelul încăperii, utilizînd reieţiile de calcul din [2] .Procedura de calcul care realizează această operaţie este notată cu A în schema generală din fig.1.Schema logică de principiu a acestei proeduri este prezentată în fig.2.

Programele sînt scrise în C,putînd fi rulate pe un calculator compatibil IBM.Datorită numărului reletiv mare de date de intrare pe care trebuie să le introducă utilizatorul,s-a urmărit proiectarea unei interfeţe om-maşină cît mai prietenoase. Astfel pot fi alese culorile ecranului şi a textului,după preferinţă.Mesajele atenţionează asupra datelor care trebuiesc citite, precizîndu-se natura lor şi lungimea maximă a şirului care va fi citit,în invers video.Imediat după citire,fiecare dată de intrare este validată sau nu,după cum este sau nu de tipul aşteptat.Dacă data introdusă este eronată,se reia mesajul,pînă cînd data este corectă.

De asemeni după un grup de operaţii,utilizatorul este chestionat asupra corectitudinii rezultatelor.Dacă utilizatorul consideră că ele nu-s corecte,etapa se reia.

Programul propus a fost testat cu rezultate bune în calculul necesarului de căldură într-un atelier şcoală,într-o hală industrială şi în încăperile anexe acestora.

BIBLIOGRAFIE

1. Turbo C - Manual de utilizare
2. STAS 1907/1 - 80 Calculul necesarului de căldură
3. STAS 1907/2 - 80 Calculul necesarului de căldură
4. STAS 6472/3 - 89 Calculul termotehnic al elementelor de construcţie al clădirilor
5. ICCPDC - Instrucţiuni tehnice pentru proiectarea la stabilitate a elementelor de închidere a clădirilor - Buletimal construcţiilor nr.1 din 1990.

Sistem informatic pentru urmarirea productiei într-o sectie de tesatorie sau finisaj

Autori: c.p.III R.Romanescu, ing.C.Codreanu
ing.C.Moga I.P.A. Iasi
as.ing.A.Condorovici, I.P. Iasi

Acest sistem informatic de proces realizează culegerea automată si prelucrarea în timp real a datelor privitoare la situaţia unei secţii de ţesătorie în ceea ce priveşte starea de funţionare, gradul de încărcare al maşinilor, lucrătorilor, schimburilor, volumul global al producţiei sau cel realizat pe diverse perioade de timp, tipuri de produse finite, tipuri de maşini, pontajul automat, etc.

Introducere

O secţie de ţesătorie este formată din cîteva sute de maşini de ţesut. Fiecare maşină de ţesut este prevăzută cu un contor mecanic, care contorizează numărul de rotaţii ale arborelui principal.

Numărul de mii bătăi, pe care îl vom nota cu "m", constituie un element principal în estimarea producţiei de material ţesut. Cu ajutorul acestei mărimi, si cunoscînd constanta specifică a materialului, se poate afla cantitatea de material produsă pe durata unui anumit interval de timp.

Pînă acum, sistemul de urmarire a producţiei se baza pe preluarea manuală a datelor de pe fiecare maşină respectiv "m" bătăi, citite pe contorul propriu al maşinii la sfîrşitul fiecărui schimb de către conducătorul de formaţie.

Acesta mai are obligaţia de a consemna şi regimul de funcţionare-staţionare al maşinii. Toate aceste date se completau într-o fişă de lucru a maşinii. Aceste fişe se centralizau la nivel de secţie pe schimburi, zile şi luni şi se prelucrau semiautomat de către o grupă economică din cadrul secţiei.

Lucrarea s-a materializat în realizarea unui sistem

automatică și informatic care are ca scop trecerea evidenței primare din cadrul secției de țesătorie la prelucrarea și evidența pe calculator a tuturor datelor necesare urmăririi în timp real a producției. Aceasta va avea ca rezultat mărirea flexibilității în conducere și o mai mare mare viteză decizională.

Se urmăresc și realizează:

* cuantificarea fiecărei mașini în cadrul sistemului informatic;
* înregistrarea timpului de funcționare și staționare a fiecărei mașini; mașini;
* în cadrul timpilor de staționare, aceștia se vor explicita corespunzător cauzei ce i-a produs. Se vor avea în vedere următoarele cazuri:
  - reparații planificate, reglare războaie, lipsă materie primă, lipsă piese auxiliare, lipsă lucrători;
  - reparații accidentale (semnalizate de pe fiecare mașină în parte); parte);
  - lipsă energie.

Descrirea funcțională

Echipamentul de prelucrare va fi organizat pe două nivele:

a) calculatorul de proces;
b) interfețele specifice mașină și interfața calculatorului cu procesul.

Schema bloc a întregului ansamblu poate fi urmărită în fig.1.

Se disting următoarele blocuri funcționale:

- calculatorul de proces (1);
- interfața specifică a acestuia (2);
- linii de transmisie (3);
- interfețe specifice mașină (4);
- bloc alimentare (5) BA.

Interfața specifică a calculatorului realizează adaptarea între sistemul de calcul și linia de transmisie din punct de vedere fizic și al compatibilității nivelelor logice. Ea are drept scop realizarea protocolului de emisie-recepție a

calculatorului cu restul instalației.

Linia de transmisie realizează legătura fizică între interfața specifică a calculatorului și interfețele specifice mașină. Interfețele specifice mașină realizează preluarea locală hi memorarea datelor și protocolul de emisie-recepție cu calculatorul.

Modul de funcționare este următorul: calculatorul interoghează la un anumit moment o singură interfață specifică mașină prin emiterea adresei acesteia. Interfețele specifice mașină sînt în mod normal în starea de recepție. Interfața care își recunoaște adresa va trece ulterior în stare de emisie. Ea va transmite către interfața specifică a calculatorului (deci unitatea centrală) valoarea lui "n", reprezentînd numărul de mii bătăi, înregistrat în acel moment pe mașină, precum și semnalele de stare ale mașinii.

Preluarea datelor se va face de două ori consecutiv pentru fiecare mașină, pentru a se elimina o parte din posibilele erori apărute la preluarea primară a datelor.

Descrierea constructiva

Interfața specifică mașină

Această parte a instalației se află amplasată în apropierea mașinilor supravegheate. Pentru a proteja montajul se recomandă montarea alăturată de mașină, nu pe șasiul acesteia (supus unui regim considerabil de vibrații).

Montajul este realizat constructiv pe o monoplacă de cablaj imprimat, conținînd un sistem dedicat, cu microprocesor. Două capace fixate cu șuruburi permit accesul la ambele fețe ale cablajului. Prin bucșe izolante, fixate în ramă sînt introduse trei cabluri:

a) cablul de alimentare;

b) linia de transmisie;

c) racordul la traductoarele de pe mașină (realizat cu cablu ecranat).

Linia de transmisie va fi echipată corespunzător condițiilor concrete de funcționare.

Se recomandă montarea direct pe această cutie a comutatoarelor de stare (oprire avarie-reparație accidentală)

precum și a semnalizării optice (opțională) aferente: alimentare interfață, martori ai poziților comutatoarelor, etc.

Interfața specifică calculator

Această parte a sistemului, spre deosebire de informația specifică mașină, este unică și funcționează în condiții mult mai ușoare, alături de calculator.

Din punct de vedere constructiv, este realizată pe cablaj imprimat, montat într-o casetă.

Linia de transmisie

Această parte a sistemului are rolul de a asigura calea de comunicație între diversele părți componente. Deși din modul de alegere a protocolului de comunicație s-a urmărit reducerea riscurilor de apariție a erorilor la transmisie, se recomadă alegerea unui cablu ecranat.

Sistemul informatic de proces

Întregul volum de date privind producția de pe fiecare mașină, respectiv starea acesteia, era preluat și prelucrat manual. Aceasta duce la o puternică implicare a factorului uman, care, pe lîngă viteza redusă de operare, este și generator de distorsiuni în procesul de achiziție și prelucrare a datelor. Astfel apar de multe ori diferențe semnificative între valoarea producției indicată de contoarele de pe mașini și cea reală. La toate acestea se adaugă și faptul că în secția de producție se prelucrează materii prime de calități diferite, se produce un sortiment larg de țesături pe baza aceluiași proces tehnologic, iar cererile și livrările de produse fluctuează de la o zi la alta.

Dacă am ține cont doar de aceste considerente, se remarcă oportunitatea introducerii sistemului informatic pentru urmărirea producției într-o întreprindere textilă.

Trebuie menționate în același timp și avantajele ce apar ca urmare a acestei introduceri, avantaje ce în sistemul actual existent, de culegere și prelucrare manuală a informațiilor privind volumul de materii prime, a stării

mașinilor, a volumului producției pe sortimente, etc. nu sînt posibile. Este vorba de o informare operativă și obiectivă a factorilor umani de decizie, precum și a unor eventuale sisteme informatice de gestiune economică de la nivelele ierarhice superioare, prin elaborarea periodică sau la cerere a unor rapoarte sau informații adecvate asupra realizărilor sau stării procesului industrial (bilanțuri, situații pe ore, schimburi, zile, etc.).

Trebuie remarcat faptul că sistemul informatic de urmărire a producției poate furniza informații în timp real despre starea secției, permițînd luarea rapidă a unor decizii pentru optimizarea la maximum a procesului de producție. Niciodată nu se va putea ajunge la astfel de performanțe folosind operatorii umani care să completeze fișele fiecărei mașini, să le centralizeze, să prelucreze informațiile rezultate din aceste fișe și să le prezinte factorului decizional în timp util, ci se va crea întotdeauna un decalaj între starea reală a secției și situația cunoscută de șeful ei.

În această ordine de idei, a avantajelor nou create prin introducerea sistemului informatic, se înscrie și posibilitatea unei mai profunde cunoașteri a mecanismelor intime ale procesului industrial prin culegerea și prelucrarea unor informații precise și la timp, ceea ce permite o mai bună încărcare a utilajelor și instalațiilor, o mai bună organizare a muncii, perfecționarea modelelor și algoritmilor de conducere, etc.

Tot acest sistem informatic va mai permite evitarea apariției unor deranjamente sau avarii și limitarea consecințelor acestora, datorită posibilității investigării mai profunde a dinamicii procesului tehnologic, în special în perioadele de regimuri tranzitorii, de perturbații mari și pe baza rapoartelor post-avarie și a analizelor diagramelor elaborate automat.

Concluzii

Acest sistem informatic de proces, ce poate fi utilizat în unități productive de alt profil, conferă o mai mare flexibilitate și o mai mare viteză decizională, la un preț avantajos (rezultat al unei proiectări îngrijite).

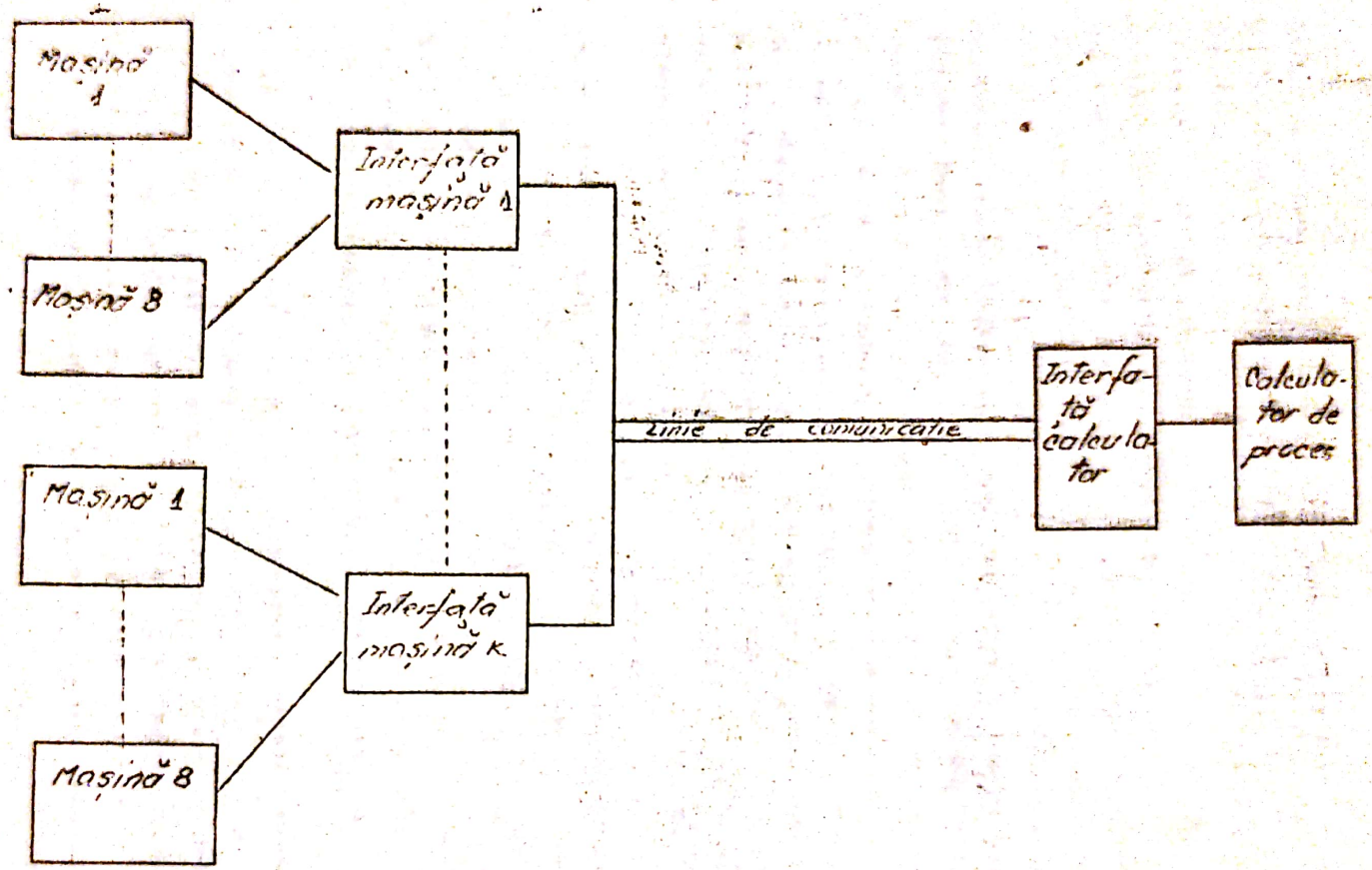


Fig. nr. 1

Sistem informatic pentru urmărirea producției într-o secție de țesătorie sau finisaj

SISTEM INFORMATIC PENTRU PROCESE TEHNOLOGICE CU ELEMENTE NUMEROASE SI TIPODIMENSIUNI PUTINE

Ing.Costescu-Gh. C-tin-Vasile
Ing.Georgescu Cristian
Universitatea "Dunarea de Jos"-Galati
Ing.Gheorghe Gabriel
Ing.Stavila Sorin
F.E.A.-Bucuresti
Ing.Vasiliu H. Zonel
R.E.N.E.L.-F.R.E. Galati

Rezumat

Lucrarea incearca sa raspunda unor greutati de patrundere a informaticii in domeniul proceselor tehnologice, greutati ridicate de faptul ca o larga gama de procese au foarte multe elemente de urmarit sau de comandat. Aceste elemente sint de citeva tipodimensiuni, fapt care produce greutati de realizare hardware si software. In activitatea Filialelor de Retele Electrice din tara se intilnesc in mod curent astfel de situatii. Ideea de baza este autonomizarea modulelor de intrari/iesiri numerice in/din calculator, prin preluarea unor functii ale unitatii centrale si amplasarea lor in teren, la distanta de acesta.Se obtine in acest fel o crestere substantiala a fiabilitatii si o scadere de aproximativ 10 ori a costului per semnalizare.

1.Introducere

In procesele tehnologice din ramura electroenergetica, exista foarte multe elemente de acelasi tip sau de citeva tipuri, respectiv aparatura de comutatie primara si/sau secundara, LED-uri sau lampi in panouri sinoptice, etc. Pentru controlul lor se pot folosi microcalculatoare de proces din gamele SPOT 80 si/sau ECAROM-881M, dar, in acest caz, controlul a circa 4000 elemente produce repede fenomenul de "saturare" a modulelor IN si/sau EN, necesitind cheltuieli mari (circa 1000 lei/element) fara a avea asigurata fiabilitatea necesara.Mai este faptul ca, nu se stie cind anume s-a defectat "controlul" unui element oarecare, astfel ca sistemul este si costisitor, si ineficient.
S-a impus deci gasirea unui sistem ieftin (100 lei/element) si eficient, in sensul ca este fiabil si totodata capabil sa semnaleze el insusi cind s-a pierdut controlul asupra unui element. Acest sistem se preteaza la aplicarea in elctroenergetica la controlul aparaturii de comutatie primara si secundara, panouri sinoptice, etc., unde elementele sint numeroase si de tipuri putine.

2.Arhitectura sistemului

Principiile care stau la baza arhitecturii sistemului sint in numar de trei:
- prelucrarea distribuita prin "spargerea" microcalculatorului in module amplasate in diverse zone ale procesului tehnologic si nu concentrate intr-un echipament unitar, legatura facindu-se la orice distanta cu sau fara MODEM;

- adresarea elementelor nu prin propria intrare sau iesire numerica, ci prin magistrale de proces separate, actionind cite un pachet de porturi, or cite un pachet de module;
- evitarea necesitatii vitezi mari de lucru prin mesaje cu frecventa mica.

Arhitectura o rezultat de la sine dupa definirea principiilor, respectiv un modul central, conectat prin serialele sale in retea de calculatoare si la propria-i periferie, modul central legat la modulele de lucru amplasate in instalatii prin cabluri jucind rol de magistrale.Aceste cabluri constituie o mica fractiune din volumul cablurilor solutiei traditionale. Fiecare modul de lucru deserveste un subansamblu de elemente, structurat pe considerente de vecinatate a elementelor sau pe considerente de grup functional al elementelor.

Modulul central are propria lui periferie si este conectat intr-o retea de calculatoare, de la care primeste comenzi, careia ii furnizeaza informatii inclusiv despre defectele de control.El are posibilitatea de a lucra si individual, fie ca asa a fost conceput, fie ca este vorba de o avarie.

3.Implicatii si rezolvari software

Functionarea sistemului este conceputa astfel incit microcalculatorul sa poata fi integrat intr-o retea, lui revenindu-i sarcina de interfata intre procesul tehnologic si sistemele de calcul. De aici au rezultat cele trei module software, respectiv: modulul pentru controlul elementelor de proces, modulul de comunicatie cu sistemele externe si modulul de lucru in regim autonom.

Modulul pentru controlul elementelor de proces este realizat sub forma unei imagini-obiect, ramas definitiv si conceput de fabricant.

Modulul de comunicatie este realizat sub forma unui program generator de imagine-obiect conform cu protocolul sistemului de calcul exterior..

Modulul de lucru autonom este conceput de beneficiar pe baza intrarilor/iesirilor precizate de foaia de insotire a echipamentului. Cu ajutorul unui fisier de comenzi indirecte, cele trei imagini-obiect sint legate, realizindu-se imaginea executabila. Beneficiarului nu-i ramine decit a tine cont de sistemul de calcul exterior, iar pentru modulul autonom de lucru, de modalitatea in care a legat modulele periferice-la elementele de proces, respectind adresa fiecarui element de proces si natura legaturii, (intrare sau iesire).

4.Realizare practica

Experimentarea s-a facut la F.R.E. Galati. Ea consta in conectarea modului-central la un minicalculator tip CORAL 4030, care ii comunica toate comenzile necesare. Modulul central, amplasat linga minicalculator, este legat la un modul de lucru amplasat la distanta de 50 m in panoul sinoptic, executind comenzi asupra LED-urilor dintr-un subpanou.

Lucrarea este in curs de dezvoltare, urmind ca doua panouri sinoptice, aflate in doua camere de comanda diferite, fiecare panou avind cite 9 subpanouri, cu un total de circa 4000 LED-uri, sa fie comandate de un singur modul central.

Acesta este amplasat linga minicalculator si comanda module de lucru realizate sub forma unor placi de marimea unui format

Scanned with OKEN Scanner

A4, montate cite 2-3 bucati pe spatele usilor fiecarui subpanou. Legatura se realizeaza prin trei cabluri cu cite 8 fire, fiecare din ele deservind cite un grup de module de lucru.

Echipamentul executa comenzile asupra LED-urilor, soneriilor si hupelor, efectuind schimbari de stare, pilpiiri cu frecvente diferite si primind comenzi manuale de la utilizator, de confirmare a pilpiirilor si anulare a soneriilor sau a hupelor.

5.Concluzii

In ramura electroenergetica exista activitati care, fara a necesita prelucrari de calcul complicate, "satureaza" echipamentele prin numarul enorm de elemente de proces. In consecinta, s-a impus necesitatea realizarii unui microcalculator dedicat, care sa joace rol de interfata intre elementele de proces si sistemele de control.

S-a avut in vedere, pe de o parte, conceptia de realizare pe baza prelucrarii distribuite a datelor si adresarea prin magistrale de proces separate cu viteza de lucru obisnuita.Pe de alta parte, s-a avut in vedere implementarea usoara hardware amplasind descentralizat modulele de lucru si identificind elementele de proces prin adresele intrarilor / iesirilor.Din punct de vedere software, se realizeaza o implementare usoara prin transparenta modulului de control al elementelor de proces, generarea ad - hoc a modulului de comunicatie in conformitate cu protocolul utilizatorului si prin posibilitatea unui modul de functionare in regim autonom.

Costul controlului unui element de proces s-a micsorat de la 1000 lei/element la 100 lei/element.

Bibliografie

[1] Manual de exploatare si intretinere ECAROM - 881M
Editat de F.E.A. - Bucuresti
[2] Murray, W. D., Computer and Digital System.
Architecture, Prentice Hall, N.J. 1990

Scanned with OKEN Scanner

ESMUL - PACHET DE PROGRAME UTILIZAT PENTRU ESTIMAREA ÎN TIMP REAL A PROCESELOR MULTIVARIABILE

Ing.D.I.Curiac, Dr.ing.O.Prostean, Dr.ing.B.Lustrea
Universitatea Tehnică Timişoara

1. Introducere

Estimarea proceselor multivariabile este un domeniu de mare actualitate în automatica modernă, stînd la baza analizei şi a conducerii adaptive a sistemelor dinamice. Dezvoltarea sistemelor de calcul şi a limbajelor evoluate a dat posibilitatea extinderii ariei de utilizare a algoritmilor de estimare multivariabilă la procese rapide.

Lucrarea de faţă prezintă pachetul de programe ESMUL utilizat pentru estimarea în timp real a parametrilor proceselor multivariabile. Estimarea multivariabilă, ca etapă decisivă în stabilirea performanţelor conducerii adaptive presupune, indiferent de algoritmul utilizat, operaţii matriciale care necesită un timp de calcul relativ mare. Aceasta depinde în principal de patru factori, dintre care doi "obiectivi" -ordinul procesului tehnic precum şi numărul intrărilor şi ieşirilor procesului- şi doi "subiectivi" -algoritmul de estimare utilizat precum şi experienţa programatorului. Utilizarea limbajului "C" a facilitat creşterea performanţelor pachetului de programe propus, asigurînd posibilitatea conducerii în timp real a proceselor avînd constante de timp limitate inferior la 1 ms.

2. Descrierea pachetului de programe ESMUL

Pachetul ESMUL are la bază algoritmul de estimare multivariabilă CMMP on-line /1/, putînd fi utilizat în trei variante ce se deosebesc prin modul de obţinere a seturilor de date intrare-ieşire.

In primele două varianta, destinate problemelor de analiză, datele sînt generate prin simularea procesului pe baza modelului său discretizat şi respectiv sînt introduse de utilizator de la tastatură. In cea de a treia variantă, destinată conducerii în timp real, datele sînt preluate direct din proces.

Pachetul ESMUL, scris în limbajul "C", este user-friendly, operatorul avînd acces la un fişier help ce descrie funcţionarea fiecărui program din componenţă. Este prevăzută totodată posibilitatea reprezentării grafice a parametrilor estimaţi precum şi calculul unor indicatori ai calităţii estimării.

In prima variantă (date simulate) pachetul ESMUL are structura, din figura de mai jos

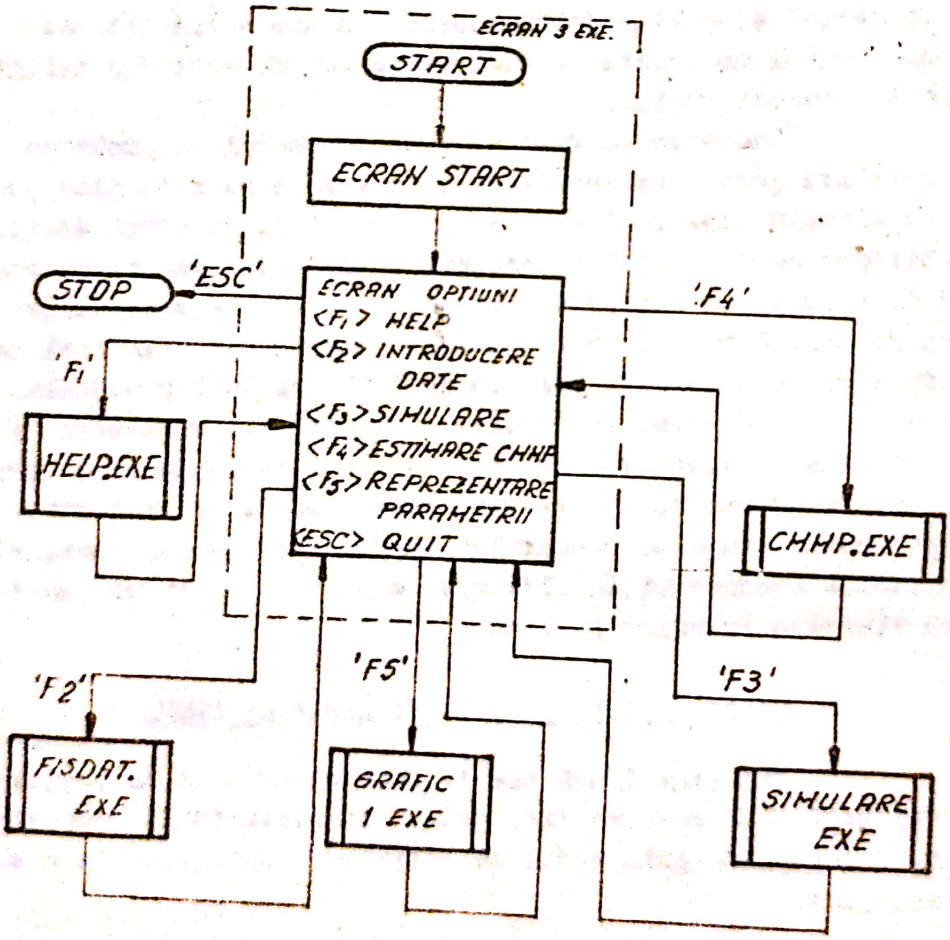


Figura 1. Structura pachetului ES MUL

Rolul central în cadrul pachetului de programe ASMUL este deţinut de programul monitor ECRAN3.EXE, care realizează interfaţa cu utilizatorul şi monitorizarea celorlalte programe.În funcţie de opţiunea utilizatorului sau în funcţie de un sistem de priorităţi definit de către acesta, pot fi apelate programele. HELP.EXE,FISDAT.EXE,SIMULARE.EXE,CMMP.EXE,GRAFIC1.EXE, respectiv abandonarea execuţiei programului şi ieşirea în sistemul de operare DOS.

Programul HELP.EXE apelat prin apăsarea tastei F1 afişează pe ecran o serie de informaţii referitoare la întregul pachet de programe, cum ar fi: rolul programelor în cadrul sistemului, iniţializarea pachetului de programe, prezentarea algoritmului de estimare multivariabilă utilizat, etc.

Programul FISDAT.EXE permite introducerea în fişierul ordin.dat a numărului de intrări şi ieşiri, a ordinului mediei alunecătoare precum şi a ordinului autoregresiei. Fişierul conţine numere întregi şi este utilizat de programul CMMP.EXE. La cererea utilizatorului acest program mai permite introducerea seturilor de date intrare-ieşire de la tastatură în fişierul intios.dat.

Programul SIMULARE.EXE permite obţinerea setului de date intrare-ieşire prin simulare. Seturile de date obţinute sînt salvate de asemenea în fişierul intios.dat.

Programul CMMP.EXE realizează estimarea multivariabilă a parametrilor prin metoda CMMP on-line. Programul primeşte ca date de intrare numărul de intrări şi ieşiri ale procesului, ordinul mediei alunecătoare, ordinul autoregresiei (fişierul ordin.dat) precum şi seturile de date intrare-ieşire. Parametrii estimaţi sînt salvaţi într-un fişier ce conţine numere flotante, param.dat.

Programul GRAFIC1.EXE permite reprezentarea grafică a parametrilor estimaţi precum şi calculul unor indicatori de calitate ai estimării.

In varianta destinată conducerii directe, în vederea creşterii performanţelor, pachetul de programe este simplificat renunţîndu-se la programul HELP.EXE. In acest caz fişierul param.dat conţine ultimele 500 de valori estimate ale parametrilor.

Prin modificările aduse, crește viteza de prelucrare a unui set de date intrare-ieșire. De exemplu, pentru un proces cu trei intrări și trei ieșiri, timpul de prelucrare se situează sub 4 ms. In conducerea adaptivă acest lucru este suficient pentru reglarea unor procese având constante de timp mai mari de 1 ms, deoarece estimarea parametrilor se face în funcție de viteza de variație a parametrilor, odată la 5 până la 30 de pași.

- Pachetul BSMUL este stocat pe o dischetă de 1.2M și necesită următoarea configurație minimă a sistemului de calcul: calculator compatibil PC-AT cu o unitate floppy de 1.2M și harddisk. Rezultate suficient de bune se obțin la o frecvență de lucru a sistemului de minim 12 MHz.

Bibliografie

1. Identificarea asistată de calculator a sistemelor - M.Tertișco, P.Stoica, Th.Popescu - Editura Tehnică București 1987.
2. Tehnici de identificare și modelare - O.Prețeanu, I.Mureșan - Curs litc I.P.T.V. Timișoara.
3. The C Programming Language - B.W.Kernighan, D.M.Ritchie - second edition - Prentice Hall Inc.
4. Structuri de date și tehnici de programare - V.Crețu curs litc I.P.T.V.Timișoara 1987.
5. Sistemul de operare DOS. funcții sistem - ediția.II-a Cluj-Napoca 1991.

C U P R I N S

D1. Controler de linie telefonica
A.Graur,D.Potorac,T.Morosan - Univ."Stefan cel Mare"Suceava
pag. 1

D2. Echipament pentru transmisiuni de date
A.Graur,D.Potorac,T.Morosanu - Univ."Stefan cel Mare"Suceava
pag. 7

D3. Interfata specializata pentru achizitii de marimi analogice
A.Graur,D.Potorac,T.Morosanu - Univ."Stefan cel Mare"Suceava
pag. 13

D4. Algoritmi si circuit dedicat de imbunatatire a eficientei procesului de corectie in transmisiile sincrone cu pondere constanta
St.Rezeanu - I.P.Iasi, pag. 19

D5. Circuit de introducere optima a conectiei de faza aferent unui sincronizator de bit digital cu urmarirea tranzitiilor
St.Rezeanu - I.P.Iasi, pag. 25

D6. Metoda neconventionala de achizitionare a informatiei despre temperatura
Nic.Galupa, V.Mayaya - I.P.Iasi, pag. 31

D7. Sinteza automatelor secventiale utilizind arbori multivalenti
L.Festila, S.Hintea - I.P.Cluj-Napoca, pag.37

D8. Procedeu de sinteza automata a circuitelor logice combina - tionale multivalente
L.Festila - I.P.Cluj-Napoca, pag.41

D9. Sisteme secventiale analiza/simulare
Fl.Hoza - I.P.Iasi, pag. 47

D10.Utilizarea inedita a timerului I8253 in comunicatii seriale
Al.Valachi, D.Olaru - I.P.Iasi, pag. 55

D11. Simulare grafica a unui planificator de actiuni pentru roboti
S.Ticrea , C.Rusu - ACOMIN, I.P.Cluj-Napoca,pag.61

D12. Arhitectura biprocesor pentru aplicatii de instrumentatie
St.Poli - I.P.Iasi, pag. 71

D13. Metode de specificare si verificare a protocoalelor in re - telele de calculatoare
K.Pusztai,V.Dadarlat - I.P.Cluj-Napoca, pag. 81

Scanned with OKEN Scanner

Arhitectura unui sistem multiprocesor de tip cub
D.Grigoras - I.P.Iasi, pag. 85
Solutii de gestiune a resurselor in implementarea structu - rilor de reglare numerica pe arhitecturi de 8 biti
Oct.Pastravanu,S.Ifrim - I.P.Iasi, pag.91
Planificarea task-urilor in sistemele de conducere a proceselor
M.Litoiu, - I.P.Bucuresti,INMT, pag.98
Alegerea elementelor de calcul in sistemele de conducere a proceselor
M.Litoiu - I.P.Bucuresti, pag.104
Sistem de criptare cu cheie publica prin metoda functiilor Unicens de tip "Rucsac"
[illegible] - I.P.Bucuresti, pag.110
Operatii geometrice asupra solidelor modelate prin octrees
D.Gorgan, D.Rusu - I.P.Cluj-Napoca, pag.118
Reprezentarea geometrica a modelelor topologice arborescente
D.Gorgan,L.Albert,D.Rusu - I.P.Cluj-Napoca, pag.125
Realismul vizual al reprezentarilor grafice. Graftali
I.Farkas - U.T.Timisoara, pag.130
Redarea umbrelor scene grafice 3D modelate prin retele de poligoane
I.Farkas,G.Farkas - U.T.Timisoara, pag.134
Metoda de recunoastere a obiectelor poligonale
S.Ifrim - I.P.Iasi, pag.138
Sistem de laborator pentru dezvoltarea aplicatiilor de con - ducere IBM-PC SPOT
[illegible],C.Radu,M.Litoiu - I.P.Bucuresti, pag.142
Pachet pentru analiza programelor de automat programabil pe bit (AP MICRO, AP 101, AP 201)
[illegible], C.Caileanu - I.P.Iasi, pag. 150
Analiza programelor pentru automatele programabile pe [illegible] (AP [illegible])
[illegible], C.Caileanu - I.P.Iasi, pag. 154
[illegible] si simularea circuitelor digitale
[illegible],D.Rusu
[illegible] - I.P.Cluj-Napoca,pag.158

E13. Asamblor pentru sisteme cu transputer T 414/T 800
C.Barsan - I.P.Iasi, pag.
L.Albulet,S.Beţenariu,M.Roman
L.Barsan - I.P.T.Iasi, pag.166
E14. Sisteme expert in comunicatie cu baze de date
D.Carstoiu - I.P.Bucuresti, pag.174
E15. Sisteme suport de decizie cu aplicatii in medicina
D.Carstoiu - I.P.Bucuresti, pag.182
E16. Analiza numerica a cimpului magnetic de excitatie la un vibrator electrodinamic
I.Vadan,S.Pedre,A.Farcas - I.P.Cluj-Napoca,pag.190
E17. Metoda de apreciere a comportarii intreruptoarelor de inalta tensiune la comutatie cu ajutorul calculatorului electronic
G.Curcenu,Gh.Floriganta - ICEMET Craiova, U.T.Craiova
M.Defta - ICEMET Craiova, pag.200
E18. Program pentru calculul necesarului de caldura pentru constructii civile si industriale
Cl.Botez - I.P.Iasi, pag.206
E19. Sistem informatic pentru urmarirea productiei intr-o sectie de tesatorie sau finisaj
R.Romanescu,C.Codreanu,C.Moga - IPA Iasi
Adr.Condorovici - I.P.Iasi, pag.210
E20. Sistem informatic pentru procese tehnologice cu elemente numeroase si tipodimensiuni putine
Gh.Costescu, C.Georgescu - Univ.Galati
G.Gheorghe,S.Stanila - FEA Bucuresti
Z.Vasiliu - REHEL-FRE Galati,pag.216
E21. Pachet de programe utilizat pentru estimarea in timp real proceselor multivariabile
D.I.Chiriac,O.Prostean
B.Lustrea - U.T.Craiova, pag.221

Scanned with OKEN Scanner